



NightFront User Guide

Non-Dominate the Night

Version 3.19.1

Contents

NightFront User Guide

- Contents of this guide

Getting started

- Installing NightFront
- First launch
- The NightFront workflow
- How a night is planned: utilization and quality
- One config file per imaging system
- Workspace tabs for several rigs
- Saving and opening configs
- The six nav-rail pages
- Active versus Planned
- The desktop app and the NINA plugin
- Your first night

Site

- Site Location
- Forecast Source
- Forecast chart
- Site Climate
- Sites Library

Imaging System

- Rig details
- Filters sub-tab
- Moon sub-tab
- Exposures sub-tab
- Variants sub-tab
- Horizon sub-tab
- Overheads sub-tab
- Loading and saving an imaging-system config

Targets

- Active and Planned targets
- Adding targets
- The target card
- Reload from disk
- Coordinates, templates and filter plans
- Mosaic cards
- Renaming a target

- Priority order
- Reviewed stamps
- Moving a target with accepted frames
- Completed targets
- The Recommender
- Saving target changes

Discover Targets

- The catalog store
- Choosing what to search
- Reading the results
- Discovery row actions

The Sky gallery

- Setting up the gallery
- The map
- Posting images
- Placing an image
- An image's card
- Zooming into the original
- Framing on the map
- Where the imagery comes from

Schedule

- How the segments fit together

Planning the season

- Seasonal Setup
- Seasonal Plan

Planning a night

- Utilization and Quality, in operator terms
- Nightly Setup
- Nightly Plan
- Editor chip

Analysis

- The night logs
- Choosing which nights
- Last night
- Overheads
- Autofocus
- Report

Processing

- The closed loop
- The Projects tree on disk

- Processing sub-tabs
- Projects sub-tab
- The rejected-frame ledger
- Completing a project
- Library sub-tab
- Processing Settings sub-tab
- PixInsight and Siril
- Online-only files

Remote processing

- Setting up a server
- Adding a peer
- What runs remotely
- While a job is away
- Collecting the result
- What's different about a remote job today

Notifications

- What sends a message
- Setting up Telegram
- Setting up Pushover
- Testing it
- Limits worth knowing

Running with NINA

- What the plugin does in NINA
- A recommended NINA sequence
- Plugin settings that shape a night
- Which config the plugin reads
- Sharing files between the desktop and the imaging computer
- Forecasts on the imaging computer
- The plugin guide
- Command-line summary

Files and data

- Your config files
- Sidecar files beside a config
- Files written by an export
- The NightFront settings folder
- Data inside your Projects tree
- Upgrades and uninstalling

Troubleshooting

- NightFront froze or closed unexpectedly
- Windows install or upgrade fails with error 1603

The ApplImage will not start
macOS says the app cannot be opened
The forecast will not load
A processing peer is unreachable
A target gets no time
Generated plans disappeared
Recent files and settings were reset
Problems inside NINA
Still stuck

Glossary

NightFront User Guide

NightFront orchestrates your astrophotography over a whole season, for all of your targets at once, from choosing which to image to deciding when each one is finished.

NightFront began as an effort to build a better target scheduler, one that plans each night for higher-quality data and more time on the sensor. It does that with search algorithms that Jeff Ridder created and used successfully on research and development programs throughout his career. That scheduler is still at the heart of NightFront, which has since grown into a complete astrophotography system.

- **Find and frame targets.** Discover Targets ranks objects against your rig, your sky and the months you choose, and frames them on the Sky map.
- **Choose the months.** The Schedule page's roadmap uses your site's real climate to show which months are best for imaging each object, and which projects fit the season ahead.
- **Plan each night.** Schedule optimizes every night for the best data your sky allows, and the NINA plugin runs it unattended on your imaging computer, replanning when conditions change.
- **See what happened.** Analysis reads NINA's own logs: last night's results against the plan, what your overheads really cost, and a scrubbed report for support.
- **Process as you go.** Processing calibrates, reviews and stacks incrementally, and lets you reject the frames that fall short, so only data you keep counts toward a project.
- **Know when you're done.** NightFront helps you decide each project is finished based on the quality of the data itself. Not a wish. Not a prayer. Data.

This guide walks through the app one screen at a time. If you're new, start with **Getting started** and follow its first-night walkthrough; after that, jump to whichever page matches the screen in front of you. The **Guide** button at the foot of the app's left-hand rail opens this guide at any time.

Questions, or something this guide doesn't cover? NightFront has a Discord server at <https://discord.gg/65ZYummSwb> — the **Community** button at the foot of the left-hand rail opens it. It is the best place to ask how other people are using something, to report a problem, or to suggest what NightFront should do next.

Contents of this guide

Start here

- [Getting started](#) — installing, the big ideas, and your first night end to end

The app, screen by screen

- [Site](#) — where you image from: location, forecast and climate
- [Imaging System](#) (**System** in the rail) — your rig: filters, moon avoidance, exposure plans, overheads, and the minimum altitude and custom horizon it sees
- [Targets](#) — your projects, their progress, and priority order

- [Discovering targets](#) — finding new objects worth imaging, and framing them on the Sky map
- [The Sky gallery](#) — a map of the sky with your own finished images placed on it, from any rig
- [Schedule](#) — the page that holds the roadmap and tonight's plan, in two groups, Seasonal and Nightly
- [Planning the season](#) — the **Seasonal** group's **Setup** and **Plan**: which projects fit the months ahead
- [Planning a night](#) — the **Nightly** group's **Setup** and **Plan**, and the **Editor**: building, comparing, editing and exporting a night
- [Analysis](#) — last night's report, overhead and autofocus recommendations and a support report, from NINA's logs
- [Processing](#) — from frames on disk back to accepted time: calibration, review and stacking
- [Remote processing](#) — sending calibration, masters and stacking to another NightFront on your LAN
- [Notifications](#) — a Telegram or Pushover message when a processing step ends (set up in **App settings**)

On the imaging computer and beyond

- [Running with NINA](#) — what the plugin does, and the command line
- [Files and data](#) — where NightFront keeps things, and what survives an upgrade
- [Troubleshooting](#) — when something doesn't behave

Reference

- [Glossary](#) — the terms this guide uses

Getting started

NightFront orchestrates your astrophotography over a whole season, for all of your targets at once: finding and framing targets, choosing the months to image them, planning each night for every project in rotation, processing the frames as they arrive, and deciding when each project has enough good data to call it done. It runs as a desktop app, and a companion NINA plugin carries the nightly planner to your imaging computer so a night can be planned and replanned unattended.

This page gets you from installing the app to a first planned night.

Installing NightFront

Installers come with each NightFront release — use the files from the release you were given. There is one per platform.

Windows installer

- Run the `.msi`. It installs NightFront for **all users** of the machine, into Program Files, with a Start-menu entry and a desktop shortcut.
- **Installing needs administrator rights, and so does upgrading.** If you run the installer from a command prompt, use an elevated one — an unelevated install or upgrade fails with error 1603. See [Windows install or upgrade fails](#).
- **An upgrade replaces the installed version in place.** Just run the newer `.msi`; there is no need to uninstall first, and you will not end up with two copies.

Linux packages

Two x86_64 builds are offered; pick one.

- `.deb` — install it with your package manager (for example `sudo apt install ./nightfront_...deb`). It installs under `/opt` and adds NightFront to your applications menu under Science.
- `.AppImage` — a single file. Make it executable (`chmod +x`) and run it. On some modern distributions an AppImage needs `libfuse2` installed; if it will not start and you would rather not install that, run it with `--appimage-extract-and-run`.

macOS disk image

- The `.dmg` is built for **Apple Silicon only** — it will not run on an Intel Mac.
- **The app is not signed**, so macOS refuses a plain double-click the first time. Copy NightFront into Applications, then **right-click the app** → **Open** and confirm the dialog. (Alternatively, `xattr -dr com.apple.quarantine /Applications/NightFront.app` in Terminal.) After that it opens normally.

- The macOS build is confirmed to launch and run the optimizer. Beyond that it is lightly tested: picking PixInsight as an `.app` does not work there yet.

Note: The NINA plugin is Windows-only, like NINA. It ships separately, with its own installer and its own guide.

First launch

NightFront opens on a splash screen, then the **Site** page of a blank **Untitled** imaging system.

Heads up: On first launch NightFront fetches a weather forecast straight away, without asking. The default source, 7Timer!, is free and needs no account. See [Site](#).

The app keeps its own settings, caches and a crash log in a `.config/nightfront` folder in your home directory — on Windows, `C:\Users\<you>\.config\nightfront\`. [Files and data](#) lists what is in there.

The NightFront workflow

NightFront runs many projects at once, each at its own stage, from idea to finished data. Each stage has its own page in the app:

1. **Find and frame targets** — [Discover Targets](#) ranks objects against your rig's field of view, your sky and the months you choose, and frames them on the Sky map. Or add ones you already know on the [Targets](#) page.
2. **Choose the months** — [Planning the season](#) uses your site's real night-time climate to show when each object is best placed and which projects fit the season ahead. Targets waiting their turn sit in the **Planned** list; the ones in rotation now are **Active**.
3. **Plan each night** — [Planning a night](#) shares each night's sky among all of your Active targets for the best data it allows, and the NINA plugin runs it unattended, replanning when conditions change. Afterwards the [Analysis](#) page reads NINA's logs to show what was actually collected against the plan.
4. **Process as you go** — [Processing](#) calibrates, reviews and stacks each project's frames incrementally as nights come home, and lets you reject the ones that fall short.
5. **Know when you're done** — only the data you keep counts toward a project, so what each target still needs is measured from the data itself. When a project's data says it's enough, you complete it and the next one moves into rotation. Not a wish. Not a prayer. Data.

Then the loop turns: what every project still needs feeds straight back into tomorrow's plan.

How a night is planned: utilization and quality

Within a night you want **more** data and **better** data. The Schedule page treats those as two explicit objectives:

- **Utilization** — how much of the night's darkness ends up as exposures. Every slew, meridian flip, dither and autofocus run counts against it, and so does idle time.

- **Quality** — how good the conditions were while those exposures were taken: airmass, blended with how well the night honors your own filter-preference rules (for example, "when seeing is above average, prefer L").

The two pull against each other. Squeezing out every minute favors fewer target switches; chasing the best sky favors more of them. So instead of one answer, the optimizer returns a **Pareto front** — a family of plans, each one the best available for its particular balance of the two. You pick the point that matches what you want from the night; NightFront starts you on a balanced compromise.

Altitude limits, your custom horizon and moon avoidance are **not** objectives. They are hard constraints: no plan ever breaks them, whatever it scores.

NightFront can also build a night without the optimizer — by strict priority order (**Greedy**), by equal shares (**Proportional**), by weighted scoring rules (**Dispatch**), or by hand in the **Editor** (a chip on Schedule › Nightly › Plan) — and every one of those plans is scored on the same two axes and drawn on the same chart. See [Planning a night](#).

One config file per imaging system

Everything about one rig lives in **one JSON config file**: the site, the forecast source, the rig's filters, moon avoidance profiles, exposure templates and variants, its horizon, its targets and their progress, and every planning setting. If you run two rigs, you keep two configs.

The config is also what the NINA plugin reads, so the file you save here is the file the imaging computer plans from.

Workspace tabs for several rigs

Each open config is a tab across the top of the window, so you can have several rigs open at once.

- A tab is labeled with the rig's **Imaging System** name, or the file name if it has none yet, or **Untitled** for a new, never-saved system.
- The + at the end of the strip offers **New Imaging System, Open...**, and your **Recent Files**. Opening a file that is already open just switches to its tab.
- × closes a tab. It is disabled while that tab's optimizer is running (its label shows an hourglass), since a run cannot be stopped part-way. Closing the last tab leaves a fresh Untitled one.
- **When you quit and relaunch, NightFront reopens every saved config that was open, and selects the one you were on.** Untitled tabs are not restored, because they have no file.
- Each tab remembers which page it was showing when you switch away and back.

Saving and opening configs

- **Save** and **Save As...** sit in the bar across the top of every page, beside the name of the file this tab is working on. They save the whole config for this tab, whichever page you are looking at. **Save** on a tab that has never been saved asks where to put it. (Before 3.11 these buttons lived at the foot of five separate pages, which meant scrolling to the bottom to reach them.)

- **Save fills in when the tab has unsaved config changes**, and the filename beside it gains • *unsaved changes*. While a config is still loading, Save stays plain rather than claiming the file is up to date — at that moment NightFront cannot yet tell.
- + → **Open...** opens a config in a new tab. The Imaging System page also has **Load Config...**, which instead loads a config **into the current tab**, replacing what it shows.
- Saving a config also writes its plan sidecars beside it (your saved plan variants and seasonal plan), and after that those sidecars update themselves as you work. See [Files and data](#).

Heads up: There is no "unsaved changes" prompt. Closing a tab or quitting the app discards edits you have not saved. Save before you close.

A few actions do save the config for you immediately, because what they change must not be lost: Moving or reframing a target (which also resets its accepted counts — see [Targets](#)), and some accept and completion actions on the [Processing](#) tab.

The six nav-rail pages

The rail down the left side of each tab has six pages, in the order you normally work through them. Below them, at the foot of the rail, the **What's new** button opens the release notes (what changed in each version), and the **Guide** button opens this user guide as a PDF. Both ship with the app. Machine-wide settings (program paths, the Projects root, the processing server and notifications) are in the **App settings** button in the title bar, not on a rail page. Its **Appearance** section switches the whole app between a dark theme (the default) and a light one; the choice applies at once to every NightFront window on this computer. The screenshots in this guide show the dark theme.

Page	What lives there
Site	Location, time zone, forecast source and API key, the site's climate profile, the forecast chart, the Sites Library
Imaging System	The rig's name and optics; sub-tabs Filters (order = filter-wheel slots), Moon , Exposures , Variants , Horizon (Min Altitude and custom horizon), Overheads (slew, flip, dither and autofocus times and switches). The rail calls this page System
Targets	Active and Planned targets, Object Lookup... , Manual Entry... , Discover Targets... (Discovery), per-filter exposure plans, priority order, completed targets, Frame on the Sky map
Schedule	Two groups. Seasonal › Setup and Plan (a 1, 3 or 6-month roadmap against your site's climate); Nightly › Setup (session, objectives, filter rules, Tonight's Feasibility) and Plan (run, Pareto chart, Gantt, summary, Export Sequence... , and the Editor chip for hand-built and generated plans) — see Planning a night
Analysis	Chips Last night (what you collected against the plan), Overheads (what NINA's logs say your slews, flips, dithers and autofocus cost, with Apply , and ways to reduce them), Autofocus (how autofocus ran, the switches and filter offsets) and Report (a scrubbed support report)
Processing	Sub-tabs Projects , Library and Settings (chips Rig , Import , Calibration , Stacking): calibration, stacking and review of the frames you collected, in PixInsight or Siril

Active versus Planned

Every target is in one of two categories:

- **Active** — in the rotation now. **Tonight's planning only ever considers enabled Active targets**, in the app and in the NINA plugin alike.
- **Planned** — on the roadmap, waiting its turn. Schedule's Seasonal > Plan looks at both Active and Planned targets, so it can tell you when a Planned project's season opens.

A target card's → **Planned** and → **Active** buttons move it between the two. A project you complete on the [Processing](#) tab leaves both lists and appears under **Completed**, where **Plan again** brings it back.

Note: A target added from Discover Targets, or created on the Sky map, lands in **Planned**, not Active — it has no exposure plan yet. Give it filters and move it to Active when it is ready.

The desktop app and the NINA plugin

The two halves share one planner and one config file.

- **The desktop app** is where you decide what the planner optimizes *for*: your site and rig, your targets and their exposure plans, the balance between utilization and quality, your filter rules and your season-scale queue. It is also where you can plan nights yourself, inspect them on a chart and Gantt, and export them.
- **The NINA plugin** runs on the imaging computer, inside NINA. At dusk its **Nightly Update** instruction either imports a plan you exported for tonight or plans the night on the spot from your config and the live sky. Later, **Replan** re-solves the rest of the night when conditions change. It counts the frames it shoots, so progress carries across unattended nights. The plugin ships the planner inside it as `nightfront-cli.exe`, so the imaging computer needs no separate install of the app.

Heads up: The app and the plugin work from the same files, so when they run on different computers your NightFront config and the plugin's data folder must live in a folder both computers share and keep in sync: Google Drive, OneDrive, Syncthing, a network share or similar. See [Sharing files between the desktop and the imaging computer](#).

See [Running with NINA](#).

Your first night

1. **Site.** On the [Site](#) page, enter your latitude, longitude, elevation and time zone. Leave **Forecast Source** at its default (7Timer! unless you enter an Astrospheric API key) and check the forecast chart fills in for the coming nights.
2. **Imaging System.** On [Imaging System](#), name the rig. Under **Filters**, list the filters your wheel holds **in wheel order**. Add a **Moon** avoidance profile or two, and an exposure template under **Exposures** so new targets can pick up a ready-made exposure plan. Under **Horizon**, check **Min Altitude (°)** and load a custom horizon if trees or buildings block part of your sky.
3. **Save.** Press **Save As...** and choose where the config lives. If NINA runs on another computer, save it in a folder both computers share and sync (Google Drive, Syncthing, a network share and so on), so the plugin plans from the same file.

4. **Targets.** On [Targets](#), use **Object Lookup...** to find an object, then give it an exposure plan: one row per filter with the sub length and how many subframes you want. Make sure it is **Active** and enabled. Save again.
5. **Check feasibility.** On Schedule › Nightly › **Setup** ([Planning a night](#)), press **Check** under **Tonight's Feasibility**. It tells you, per target and filter, whether each can actually be imaged on each forecast night — and why not, if not — without running anything.
6. **Plan.** Switch to **Plan** and press ► **GA**. When it finishes, the Pareto chart shows the front and a balanced plan is selected; click other points to compare them in the Gantt and summary. ► **Greedy**, ► **Proportional** and ► **Dispatch** add their own plans to the same chart in seconds.
7. **Hand it to NINA.** Either:
 - press **Export Sequence...** and choose the plugin's data folder — Nightly Update imports that plan tonight; or
 - let the plugin plan the night itself: point its **NightFront Config (for online planning)** setting at the config you saved, and Nightly Update generates a plan at dusk.

Both routes are described in [Running with NINA](#).

Site

The **Site** tab describes the physical place you observe from: where it is on the Earth, what the weather is forecast to do over the next few nights, and how clear the sky usually is month by month. Everything NightFront plans rests on these numbers, so this is the first page to fill in for a new rig.

The page has five parts, top to bottom:

- **Site Location** (left) — name, coordinates, elevation, time zone, the forecast source and the Astrospheric API key.
- **Site Climate** (right) — twelve monthly clear-sky percentages, fetched from NASA POWER.
- **Forecast** — a chart of cloud, transparency and seeing for the nights ahead.
- **Sites Library** — save this location once and reuse it for every rig that observes from it.
- **Save / Save As...** — in the bar at the top of the window, not on this page: it writes the whole config file, exactly as it does from every other tab.

Min Altitude and the custom horizon are set per rig, on the [Imaging System](#) tab.

Site Location

Field	What it does
Load from Sites Library ▼	Copies a saved site into this rig. Once a site is linked, the button shows that site's name instead. See Sites Library .
Site Name (optional)	A label for the location. It becomes the default name when you save the site to the library.
Latitude	Degrees, arcminutes and arcseconds. The N / S button in front sets the hemisphere.
Longitude	Degrees (up to 180), arcminutes and arcseconds. The E / W button sets the side of Greenwich — there is no minus sign to type.
Elevation (m)	Your site's height above sea level in meters. It is recorded with the site; it does not change the forecast request.
Timezone (IANA, optional)	A named time zone such as <code>America/Denver</code> or <code>Europe/London</code> .

Under the latitude and longitude boxes, a small line shows the same position in decimal degrees, so you can compare it against a map or your NINA profile at a glance.

About the time zone. Enter a real IANA zone name, not a UTC offset. A named zone knows your daylight-saving rules, so night boundaries and the times on every chart stay right across a DST change. If you leave the field blank, or type something NightFront doesn't recognize, it falls back to an estimate from your longitude that has **no DST awareness**. For a name it doesn't recognize, the field turns red and the helper text under it says so. A new config starts with your computer's own time zone filled in.

Forecast Source

NightFront plans each night against a weather forecast for your site. Two live sources are available, chosen from the **Forecast Source** dropdown:

Choice	What you get
Default (Astrospheric with a key, otherwise 7Timer!)	Uses Astrospheric if an API key is entered below, and 7Timer! if not. This is what a new config uses.
7Timer! ASTRO — free, no key, 72h	Always 7Timer!, even if you also have an Astrospheric key entered.
Astrospheric — needs an API key, 82h	Astrospheric. Without a key, NightFront quietly uses 7Timer! instead.

Under the dropdown, a caption tells you what is **actually** in use — for example "*Using 7Timer! ASTRO. Covers about 72h.*" Read it after changing anything here: it is the only place that shows when your choice and the source in use differ (Astrospheric selected with no key, for instance). When 7Timer! is in use, a small attribution line credits it under that caption.

7Timer! ASTRO forecast source

[7Timer!](#) is a free, keyless, worldwide astronomy forecast derived from NOAA's GFS weather model. It is the only free source that provides all three things NightFront plans with: cloud cover, seeing and transparency.

- Its data comes in **3-hour steps** and reaches about **72 hours** ahead.
- NightFront shows it at that native resolution rather than smoothing it into hourly values it doesn't really have.
- A new install with no API key uses it automatically, so the very first launch makes a network request for your site's forecast.

Astrospheric forecast source

[Astrospheric](#) is a paid service (you need a Pro account and its API key). Its forecast reaches about **82 hours** ahead, so it usually covers one more plannable night than 7Timer!.

Enter your key in **API Key (optional)** under **Astrospheric API Key**. The key is hidden as you type; **Show / Hide** toggles it. It is saved in your config file and, if you save the site, in your Sites Library.

Astrospheric bills per request. The forecast **Refresh** button shows the price before you click it — *Refresh (65 credits)* — and only for Astrospheric; a free source shows plain *Refresh*.

Heads up: Typing an API key, or editing the latitude or longitude by hand, does not re-fetch a forecast that is already on screen. Click **Refresh** afterward. Changing the **Forecast Source** dropdown, loading a site from the library, or opening a config clears the chart and fetches again on its own. A site at a different location is a new forecast request: free from 7Timer!, a billed call from Astrospheric.

How long a forecast is kept

A fetched forecast is saved on disk and reused for a while, so reopening the app, switching tabs, or an unattended replan in NINA doesn't buy the same forecast again:

Source	Reused for
7Timer! ASTRO	3 hours (its data only changes in 3-hour steps)
Astrospheric	6 hours (about how often Astrospheric refreshes its model)

The two sources keep separate copies, so switching back and forth between them doesn't throw either one away. **Refresh** always fetches a fresh forecast, ignoring the saved copy.

When a forecast will not load

NightFront **never makes up weather**. If the forecast can't be fetched, it tells you instead of planning against an invented sky.

- **In the app:** the error appears in red under the forecast banner, and the chart reads "*No forecast data. Click Refresh.*" Fix the connection (or the key) and click **Refresh**.
- **In an unattended NINA run:** when the plugin plans or replans and the fetch fails, NightFront falls back to the last saved forecast for your site **if it is no more than 48 hours old**, and reports that the plan rests on a stale forecast rather than a live one. If there is no saved forecast, or it is older than 48 hours, no new plan is made and the plugin says why. See [Running with NINA](#).

Forecast chart

The **Forecast** section shows the forecast for your site as three horizontal bars running left to right through time.

- The **status banner** at the top names the source and how old the saved copy is (for example "*7Timer! ASTRO — cached 1.4h ago*"), with the **Refresh** button beside it.
- **Cloud**, **Transparency** and **Seeing** each get a bar. Every cell is one forecast step (an hour for Astrospheric, three hours for 7Timer!). The color runs from **white (worst)** to **deep blue (best)** — for all three bars, darker is better.
- **Seeing now:** under the bars names the seeing category for the step nearest the current time.
- A thin strip below the bars is **amber where the sun is up** and dark at night, so you can see which parts of the forecast actually matter.
- The time axis marks each new day with its date and adds an hour label roughly every six hours, in your site's time zone.
- A thin **red line** marks the current time when it falls inside the forecast.

The nights NightFront can plan are the nights this forecast covers. With 7Timer!'s 72 hours that is typically one night fewer than with Astrospheric. See [Planning a night](#).

Site Climate

A forecast only reaches three or four nights ahead. Planning over months needs something else: how clear your sky **usually** is in each month. That is what **Site Climate** holds — twelve boxes, **Jan clear %** through **Dec clear %**, each the long-run percentage of clear sky for that month.

It is used by [Planning the season](#) to estimate how many clear hours a multi-month roadmap is likely to get, and by [Discover Targets](#) to weigh how usable each candidate's months really are. The nightly plan does not use it; that uses the forecast.

Fetching from NASA POWER

Click **Fetch/Refresh from NASA POWER** to fill the twelve months from [NASA POWER](#)'s satellite-derived cloud data for your coordinates.

- NightFront pulls **10 years of hourly cloud data** (the ten most recent complete calendar years) and averages only the hours that fall inside each night's **astronomical darkness** at your site. Daytime cloud — afternoon convection that clears by dusk, for instance — doesn't drag the numbers down.
- Because that is a lot of data, it takes a while. The button reads *"Fetching... (this pulls 10 years of hourly data, may take a bit)"* until it's done.
- The caption beside the button shows where the numbers came from, for example *"Source: NASA POWER (nighttime avg, 2016–2025) (fetched 2026-09-14)"*. For a site that has never been fetched it reads *"Source: Not fetched"* and every month shows 50.
- A failed fetch shows *"Climate fetch failed: ..."* in red.
- The result is also kept on disk for your coordinates, so opening another config at the same site that has no climate numbers of its own shows it without fetching again.

Correcting a climate month by hand

You know your site better than a satellite average does. If a month is wrong — a monsoon dip the average smooths over, a reliably foggy November — type your own percentage into that month's box.

Heads up: A month you type becomes a **hand-set value that stays put**. Clicking **Fetch/Refresh from NASA POWER** again updates the fetched numbers underneath, but every month you have typed into keeps your value. There is no button to undo a single month's override; type the fetched value back in if you change your mind.

Values are clamped to 0–100.

Sites Library

If you run more than one rig from the same place, define the site once. The **Sites Library** is shared by every config you open on this computer.

A saved site holds the **site name, latitude, longitude, elevation, time zone, the Astrospheric API key and the climate profile** (fetched values and your hand-set months). It does **not** hold the forecast-source choice or the rig's altitude floor — those stay with each rig's config.

Control	What it does
Save	If this rig is linked to a saved site, asks to overwrite that site with the current values (Overwrite / Cancel). If not linked, works like Save As...
Save As...	Saves the current values as a new site under a name you give, and links this rig to it.
Manage Sites...	Opens Manage Sites , where you can Rename or Delete any saved site.

Loading a site with **Load from Sites Library** ▾ — on this tab, or from **This rig is located at:** on the [Imaging System](#) tab — **copies** the site's values into this rig. It is not a live connection: editing the site fields here doesn't change the library entry until you **Save**, and changing the library entry doesn't change this rig until you load it again.

Under the buttons, a status line keeps you honest about the link:

- *"Linked to 'Home Observatory'."* — the rig matches the saved site.
- *"Linked to 'Home Observatory' — local values have diverged from the saved site."* — you've edited something here since. **Re-sync from Sites Library** throws away the local edits and reloads the saved values; **Save** pushes your edits into the library instead.
- *"Linked site '...' no longer found in your Sites Library."* — the site was deleted (perhaps from another tab). **Save as new Site** puts the current values back in the library; **Unlink** forgets the link.

Imaging System

The **Imaging System** tab describes one rig — a telescope and camera — and how it exposes: the filters in its wheel, the moon rules that gate each filter, reusable exposure plans, the shorter-sub alternatives NINA can switch to in wind, and how low the scope can point.

One config file is one imaging system. The name you give it here is also the label on its workspace tab.

The page has three parts:

- **Rig details** — a collapsed section at the top holding the rig's name, telescope, camera, optics and site.
- Six sub-tabs: **Filters**, **Moon**, **Exposures**, **Variants**, **Horizon** and **Overheads**.
- **Load Config...** and the **Current file** line at the bottom. **Save / Save As...** are in the bar at the top of the window, shared by every page.

Rig details

The rig's identity is set once and rarely touched again, so it starts **collapsed** to give the sub-tabs the room. The header still names the rig — for example "*Rig details — Backyard RC8 · RC8 · ASI2600MM · Home Observatory*" — so you don't have to open it to see which rig this is. Click the header to expand it.

Field	What it does
Imaging System Name	Identifies this rig and becomes its workspace tab label.
Telescope	A label for the telescope. Name only.
Camera	A label for the camera. Name only.
Optics	Focal length, aperture, reducer and sensor size — see below.
Site — <i>This rig is located at:</i>	Picks the saved site this rig observes from. It copies that site's location, time zone, API key and climate into this rig, exactly like Load from Sites Library ▾ on the Site tab.

Note: Imaging System Name takes effect when you press Enter or click away from the field, not on every keystroke. Processing settings kept for this rig follow the rename, so finish typing the name before you leave the field.

Optics for mosaics and discovery

Fill these in if you shoot mosaics or want to use [Discover Targets](#). Scheduling and export don't use them.

Field	Unit
Focal length (mm)	millimeters
Aperture (mm)	millimeters
Reducer / extender (×)	a multiplier; blank means none (1×)
Sensor width (px)	pixels
Sensor height (px)	pixels
Pixel size (μm)	micrometers

Enter the sensor as pixels and pixel pitch — the numbers on the camera's datasheet and in NINA's equipment profile. Everything else is worked out for you and shown in a line underneath, for example *"23.5 × 15.7 mm sensor · f/7.2 · field 0.92° × 0.62° · 0.53"/px"*.

Until focal length, both sensor dimensions and pixel size are filled in, the line instead reads *"Enter focal length, sensor size in pixels and pixel size to enable mosaic creation."* Without optics, creating a mosaic on the [Targets](#) tab is unavailable and Discover Targets won't search. Aperture is only needed for the f-ratio.

They also let NightFront check an older mosaic. At a rotation of 45°, 135°, 225° or 315°, a mosaic laid out mirrored by an older NightFront and one laid out as NINA does fit the same panels, with rows and columns swapped. Only the camera tells them apart: the ratio of **Sensor width (px)** to **Sensor height (px)** for a grid (or, when **Focal length (mm)** and **Pixel size (μm)** are filled in too, the shape of the field of view they give, which is what NightFront lays mosaics out with), and, for a single row or column, the field of view itself, which needs all four. Without them the Targets page names such mosaics and asks for these fields, and the NINA plugin's whole-mosaic Frame refuses them the same way. The plugin can fall back on NINA's connected camera, but the numbers here win whenever they are filled in, so a wrong entry here is what it uses. See [Mosaics laid out mirrored against NINA](#).

Filters sub-tab

Filter Definitions lists every filter this rig can use. The list starts with the classic seven — **L, R, G, B, Ha, OIII** and **SII** — and you can add any others: an OSC "filter", a dual-narrowband like **L-eXtreme**, photometric filters, anything.

Click a filter to edit it; **+ Add Filter** adds a new one (named *New Filter*, role *Other*) and opens it; **X** deletes one.

Field	What it does
Name	The filter's name. Use the same spelling as the filter in NINA's filter wheel settings — that is how the NINA plugin finds the filter when it runs your plan.
Role	Luminance, Broadband, Narrowband, Osc, Photometric or Other .
Color hex (#RRGGBB)	The color of this filter's bars on the Gantt charts, e.g. #FF8800 . A swatch beside the field previews it. Leave blank for a default color.
AstroBin filter	This filter's number in AstroBin's equipment database, used in the AstroBin acquisition CSV every stack writes. Type the number, or paste the address of the filter's AstroBin page — NightFront keeps just the number. Leave blank if you don't upload to AstroBin.

Finding a filter's AstroBin number. Click **Find on AstroBin** beside the field to open AstroBin's filter list in your browser, search for your filter there, and open its page. The page's address looks like `https://app.astrobin.com/equipment/explorer/filter/4531/...` — copy the whole address into the field (or just 4531) and press Enter or click away. Text that isn't an AstroBin filter number or page address is refused with a message under the field, and the stored number stays as it was. You only need to do this once per filter; like the rest of this page, it is saved with the imaging-system config.

What the role is for. A filter's name says *which* filter it is; its role says *what kind*. NightFront uses the role where the kind matters, regardless of what you've named the filter:

- **Narrowband** and **Osc** filters tell [Discover Targets](#) that this rig can cut through sky glow on emission nebulae, which raises how reachable those objects score.
- [Planning the season](#) charts narrowband time separately from broadband time, using the role.

Set the role correctly for any filter you add — a dual-narrowband filter named `Dual-NB` is **Other** until you say it's **Narrowband**.

Filter order is your filter wheel

Heads up: The **order** of this list is your filter-wheel order. The number to the left of each filter is its **wheel slot**, and it is what an exported NINA plan tells the wheel to move to. Slots are numbered from **0** — the first filter is slot 0.

Use ↑ and ↓ on each row to arrange the list to match your physical wheel. The arrows are grayed out at the top and bottom of the list.

Getting this wrong is quiet: the NINA plugin finds filters by **name** when it imports a plan, so a mismatched order usually goes unnoticed. It matters when an exported plan file is loaded directly into NINA without the plugin. Reordering changes only slot numbers in exported plans — it doesn't change how anything is scheduled.

Heads up: Every filter used by a target must exist in this list. If a target uses a filter that isn't here, **export fails** with a message naming the filter — NightFront won't guess a wheel slot. Add the filter here and export again.

Renaming a filter here does not rename it on targets or templates that already use it. Change those to the new name too, or re-add the old name.

Moon sub-tab

Moon Avoidance Profiles decide how far from the moon each filter may point. A profile is assigned per filter on each target (and in exposure templates), so a narrowband filter can image close to a bright moon while luminance keeps well away.

Moon avoidance is a **hard rule**, not a preference: at any moment a filter that is too close to the moon simply can't be scheduled. Its target can still be imaged through its other filters if they're far enough away.

A filter with **no profile** uses a flat minimum separation instead — 30° unless your config file states otherwise — whatever the moon's phase or altitude.

The  beside each profile's **X** copies it as "Copy of ...", directly under the original and selected ready to edit — the quick way to build a second profile that differs in one number. ⚠️ One thing the copy does *not* inherit is its **Default for filters** list. Those claims are first-come-first-served, so a copy claiming the same filters could never actually be used: it would sit there looking as though it applied while the original won every time. Assign the copy whichever filters you want it to claim.

How a moon profile works

Each profile describes a curve of *required separation* over the lunar month:

- **Sep (°)** — the required separation at **full moon**, the peak of the curve.
- **Width (days)** — how quickly the requirement falls away from full moon. At **Width** days before or after full moon, the requirement is **half** of Sep. A larger width keeps you away from the moon for more of the month; a smaller one lets you back in sooner.

The requirement never quite reaches zero, even at new moon. With the default 90° and 14 days, for example, the requirement is 90° at full moon, 45° two weeks either side of it — which is about new moon — and about 72° a week from full.

Relaxing when the moon is low. A moon near or below the horizon lights the sky far less than one high overhead. The last three fields let the requirement ease off as the moon sinks:

- **Relax Scale (0–10)** — how many degrees of separation the requirement drops for every degree the moon is below **Max Alt**. **0** turns relaxation off: the moon counts fully even when it is below the horizon.
- **Max Alt (°)** — the moon altitude at and above which there is no relaxation.
- **Min Alt (°)** — the moon altitude at which the requirement has faded away entirely. Between Max Alt and Min Alt the width also narrows steadily, so the requirement concentrates ever more tightly around full moon.

For example, with Sep 90°, Relax Scale 5, Max Alt 5° and Min Alt –15°: with the moon 5° below the horizon the full-moon requirement is 40° ($90 - 5 \times 10$) over a half-as-wide window, and with the moon at –15° there is effectively no requirement at all.

Both altitudes are the **moon's** altitude, not your target's.

Moon profile preview

Under the editor, a small chart plots the profile across a lunar month from **New moon** through **Full** to **New moon**:

- The **bright blue curve** is the requirement with the moon high (at or above Max Alt).
- A **faint blue curve**, shown only when Relax Scale is above 0, is the requirement with the moon halfway between Min Alt and Max Alt.
- A thin **yellow line** marks full moon.

Each row in the profile list summarizes its numbers, for example "*sep=90° w=14d relax=0*". + **Add Profile** creates *New Profile* with Sep 90°, Width 14 days, Relax Scale 0, Min Alt –15° and Max Alt 5°.

Default moon profile for filters

Under **Default for filters**, click the filter chips this profile should be given automatically. When a new filter row is created for a target — added by hand, or loaded from an exposure template that doesn't name a profile itself — a row for one of these filters gets this profile.

- It applies **when the row is created**. Changing the chips later leaves existing targets alone.
- A template's own choice of profile wins; the default only fills a gap.
- If more than one profile claims the same filter, the first profile in the list wins.

Heads up: Targets and templates refer to a profile **by name**. If you rename or delete a profile, filters still pointing at the old name quietly fall back to the flat minimum separation when NightFront plans. Re-pick the profile on those filter rows after a rename.

Exposures sub-tab

Exposure Templates are reusable per-filter plans — "LRGB 300s", "SHO 600s" — so you don't type the same rows into every new target. + **Add Template** creates one and opens it;  copies one as "Copy of ..." with all its filter rows, directly under the original and selected ready to edit; ✕ deletes one. + **Add Filter** adds a row (starting at L, 36 frames of 300 s, gain 0, offset 10) and the ✕ at the end of a row removes it.

Column	What it means
Filter	Which filter, from your Filters list.
Planned	How many frames you want in total.
Exp(s)	Sub-exposure length in seconds. NightFront always uses exactly this value.
Gain	Camera gain for this filter.
Offset	Camera offset for this filter.
Readout	Camera readout mode — see Readout mode column . Blank means the camera's default.
Dither	Dither after every this-many frames of this filter. 1 dithers between every frame.
Moon Profile	The moon avoidance profile for this filter, or — none — for the flat minimum.

To use a template, open a target on the [Targets](#) tab and choose it from **Load template:**.

Heads up: Loading a template **replaces** the target's filter rows with a copy of the template's, starting from zero accepted frames. After that the target and the template are independent — editing or deleting the template later changes nothing on targets already created from it.

Readout mode column

Many cameras offer several readout modes (for example a high-gain or low-noise mode). The **Readout** column takes the camera's mode number as NINA lists it.

- **Blank** (the field shows *default*) means *use the camera's own default readout mode* — never mode 0. Typing `default` does the same.
- A number selects that mode. NightFront can't check it against your camera, so copy the number from NINA.
- The same column appears on each target's filter rows.

If no filter in a night states a readout mode, the exported plan contains no readout-mode instructions at all. When a night does state one, NightFront sets the mode before the frames that need it and returns the camera to its default for filters that don't. A plan like that needs the NightFront plugin to run it: loaded straight into NINA without the plugin, it fails NINA's validation.

Variants sub-tab

Exposure Variants are named, rig-wide alternatives to your normal sub lengths — say a "High Wind" variant that shoots narrowband at 300 s instead of 600 s, so a gust ruins less.

+ **Add Variant** creates one; click it to edit. For each variant:

- **Variant name** — the name you'll select in NINA.
- + **Add Filter Override** adds a row of **Filter** and **Exp(s)**, the sub length to use for that filter while the variant is active.
- *Filters you don't list here keep their normal exposure.*

The list shows each variant's overrides at a glance, such as "*Ha 300s, OIII 300s*", or "*no overrides*".

How a variant gets used. The desktop app never applies one — every plan you make in NightFront uses your normal exposures. A variant is chosen at run time by the NINA plugin's **Replan** instruction, which has an **Exposure variant** field. The usual setup is a Wind Speed Trigger that runs a Replan naming your variant when the wind rises, and a plain Replan when it drops. See [Running with NINA](#).

- **Total integration time doesn't change.** A variant re-divides the same sky time into more, shorter frames; it doesn't cut how much time the target gets.
- The name is matched **ignoring upper and lower case**.
- If the name in NINA doesn't match any variant in the config, the replan still runs — at your normal exposures. The plugin's editor marks an unrecognized name, but an unattended night gives no other sign, so check the spelling when you set it up.

Horizon sub-tab

The **Altitude Floor** decides how low this rig can usefully point. It is set per imaging system, not per site, because what a scope can see depends on where it stands. Two rigs at the same site can have very different horizons: in one observatory, one pier may be clear all the way down to the south while the other looks past a wall or the roof line.

Min Altitude floor

Min Altitude (°) is a single floor for the whole sky, from 0° to 90°; a new config starts at 20°. NightFront won't schedule a target while it is below this altitude.

Custom horizon file

A custom horizon is a per-direction floor — the trees, roofline and hills you actually see from this scope. NightFront reads the same **.hrz** file NINA uses, so the planner and your sequence agree about what is up.

Control	What it does
Load .hrz...	Opens a NINA horizon file and makes it this rig's horizon.
Reload	Reads the same file again — use it after editing the file.
Clear	Removes the custom horizon and goes back to Min Altitude.

Once loaded, a line shows the file name, how many points it has and its altitude range, for example *"observatory.hrz — 24 points, 8.0°–31.5°"*.

Heads up: A custom horizon **replaces** Min Altitude — it doesn't combine with it. In every direction, the horizon's own altitude is the floor, even where it is **lower** than Min Altitude: a horizon below the flat minimum opens that direction up, letting the scope go lower there than Min Altitude would. While a horizon is loaded, **Min Altitude (°)** is grayed out and marked *"superseded by the custom horizon"*; its value comes back into force when you **Clear** the horizon.

Things worth knowing:

- Between the points in the file, the floor is interpolated in a straight line, just as NINA does.
- The horizon is **copied into your config** when loaded. Editing the `.hrz` file afterward changes nothing until you click **Reload**.
- A file that can't be read, or has fewer than two points, is refused with *"Horizon load failed: ... — the previous horizon (if any) is unchanged."*
- When the NINA plugin plans or replans, the horizon saved in this config **wins** over the one the plugin passes along from NINA's own profile. NINA's horizon is used only when the config has none, so a mid-night replan never switches to a different sky than the one you planned against.

Overheads sub-tab

Imaging system overheads are what your rig really spends, in time, on things that are not exposing. They are properties of the imaging system *as configured on the imaging computer*, which is why they live here rather than with the planner's tuning on Schedule › Nightly › Setup (which keeps a one-line pointer to this sub-tab). They cost real time in every plan, so set them to what your rig actually takes.

Field	Default	Meaning
Max Slew/Center/Rotate (min)	5	The worst case : a move right across the sky with a big rotation change. Nearby moves are charged less — see below.
Meridian Flip (min)	5	Time a meridian flip takes.
Dither (min)	1	Time per dither.
Min Exposures Before Filter Change	1	How many frames to take in a filter before switching (at least 1).
Periodic, every N hours	off	The planner assumes NINA refocuses on this timer and budgets the time for it. It is a prediction only: NightFront does not export an autofocus trigger, so set the real one in NINA.
On every filter change	off	Run autofocus whenever the filter changes.
After every slew	off	Run autofocus after each slew to a new target.
Autofocus Run (min)	2	Time one autofocus run takes.
Periodic Interval (hrs)	1.0	The timer for periodic autofocus; rounded to a tenth of an hour when you press Enter or leave the field.

The panel scrolls inside the sub-tab like the others. The fields are saved with the config, as before.

How long a slew is actually charged

Max Slew/Center/Rotate is an upper bound, not a flat rate. It is what a slew costs at its worst: a wide move across the sky *and* a large rotation change, with centering and the rotator both running all the way to tolerance. NightFront charges less when the real move is smaller:

The move	Charged
A big move with a big rotation change	the full time you set
Rotation changes by less than 10°	half
Less than 10° across the sky	half
Both — a hop between two panels of a mosaic	a quarter
Back onto the target the mount is already on	nothing

The first slew of each night always costs the full time: the mount is parked, and NightFront cannot know where it is pointing.

Autofocus is **not** scaled this way. It depends on temperature and how long it has been, not on how far the mount moved, so an autofocus after a short hop costs the same as after a long one.

Setting overheads from your NINA logs

Instead of guessing the overhead fields, you can let your own rig's history set them. Under each field above, a one-line hint shows what NINA's log files suggest (for example *logs suggest 0.4 min (from 25 dithers, since 1 Oct 2026) — now 0.5 min*) with an **Apply** button that changes only the open config; **Save** keeps it. **See Analysis › Overheads**

› jumps to the full table. The saved night logs, reading them, the analyses and the eras all live on the [Analysis](#) page, not here. Until the NINA plugin has saved a night's log the fields behave exactly as they always have: you type the values.

Loading and saving an imaging-system config

At the foot of the page:

- **Load Config...** opens a config file (`.json`) into this tab. A status line reports "*Loaded ...*" or why it failed.
- **Current file** shows which file this tab is working on, or *None* for a config that has never been saved.
- **Save** (in the top bar) writes to the current file, or asks for a name if there isn't one yet; **Save As...** always asks.

Targets

The **Targets** tab is your project list: every object this imaging system is working on, what each one still needs per filter, and which of them tonight's planners are allowed to touch. Everything here belongs to the config that is open — one config per rig — and ordinary edits stay in memory until you **Save**.

The screenshot shows the 'Targets' tab in a software interface. The interface is divided into two main sections: 'Active' and 'Planned'. The 'Active' section on the left shows a list of targets with columns for Filter, Planned, Accepted, Remaining, Plan (N), Acc (N), Rem (N), Exp (s), Gain, Offset, Readout, Other, and Moon Profile. The 'Planned' section on the right shows a list of targets with columns for Display name, RA, Dec, and a progress bar. The 'Active' section also includes a 'Rotation Angle (°)' field and a 'Load template' dropdown. The 'Planned' section includes a 'Reset to plan order' button. The interface also features a sidebar with navigation options like 'Sky', 'Site', 'System', 'Targets', 'Schedule', 'An', 'Analysis', 'Pr', and 'Processing'. The top of the interface has a 'Save' button and a 'Save As...' button.

Filter	Planned	Accepted	Remaining	Plan (N)	Acc (N)	Rem (N)	Exp (s)	Gain	Offset	Readout	Other	Moon Profile
Ha	100	92	8	16.7	15.3	1.3	600	2750	0	default	1	H&R
OH	150	132	18	25.0	22.0	3.0	600	2750	0	default	1	OH
R	12	12	0	1.0	1.0	0.0	300	0	0	default	1	LRGB
G	12	12	0	1.0	1.0	0.0	300	0	0	default	1	LRGB
B	12	12	0	1.0	1.0	0.0	300	0	0	default	1	LRGB
Total				44.7	40.3	4.3						

Active and Planned targets

The page shows two lists side by side (on a narrow window they collapse into one column with an **Active / Planned** switcher at the top):

- **Active** targets are in the rotation now. They are the only targets the nightly planners — the genetic optimizer, Greedy, Proportional and Dispatch — and the NINA plugin will schedule.
- **Planned** targets are on the roadmap, waiting. Schedule's Seasonal > Plan considers them; tonight's plan never does.

Move a target between the two with the → **Planned** or → **Active** button on its card. On a mosaic, that button moves every panel together.

A target is scheduled tonight only when it is **Active**, its checkbox is ticked (enabled), and at least one of its filters has time remaining. A target with nothing remaining is quietly left out of the plan, not shown as a failure.

The checkbox belongs to Active targets only: it means "eligible for nightly planning and for the NINA plugin's planning". Planned cards don't show it, and the [seasonal plan](#) ignores it, so a target left out of tonight still appears

in the season roadmap. An unticked Active target stays unticked across saves, reloads and nights until you tick it, and moving a target from Planned to Active ticks it.

The **Planned** column is listed in seasonal-plan order (soonest start first) once you have run a seasonal plan, and in the order targets were added before that. The **Active** column follows the same order until you set a priority — see [Priority order](#).

Below the two lists, a **Completed (N)** strip appears once any project has been marked complete — see [Completed targets](#).

Adding targets

Three of the four buttons across the top of the page are three equal ways to get a target onto the list. (The fourth, **Reload from disk...**, is described [below](#) and adds nothing.)

Object Lookup

Object Lookup... opens **Object Lookup (SIMBAD)**. Type a name SIMBAD knows — NGC 7293, M42, Helix Nebula — and press **Search** (or Enter). The result shows SIMBAD's main identifier, coordinates, type, magnitude and size. **+ Add to Plan** adds it as an **Active** target.

- The dialog stays open after adding, so you can look up several objects in a row. **Close** when you are done.
- An object already on the list under the same SIMBAD identifier is not added twice.
- SIMBAD is consulted once, when you add the target. After that the stored coordinates are what NightFront uses; it never re-resolves a target behind your back.

Manual entry

Manual Entry... is for anything SIMBAD does not know, or a pointing you worked out yourself. It has two modes:

Single target — a **Name / Label**, an optional **Type** (defaults to `USER`), and RA and Dec in sexagesimal fields. The decimal-degree equivalent is echoed underneath so a transposed digit is easy to spot. **+ Add** creates an **Active** target and closes the dialog.

Mosaic — NightFront lays out the panels for you from your rig's field of view:

Field	Default	Meaning
Mosaic name	—	Panels are named " <i>name</i> Panel 1", " <i>name</i> Panel 2", ...
Centre RA / Centre Dec	—	The center of the whole mosaic
Rotation Angle (°)	0	NINA position angle, east of north (the field's hint) — the Position Angle of NINA's sequencer target, which the Framing Assistant's Rotation box also shows; every panel gets it
Panels across / Panels down	2 / 2	The grid
Overlap (%)	20	Overlap between neighboring panels, 0–99

A preview line shows the panel count, each panel's size, the total coverage, and roughly how much dark time one full round of panels costs. + **Create Mosaic** adds all panels as **Active** targets, each with a 30-minute minimum imaging time.

Note: Mosaic mode needs optics. Set focal length, sensor size in pixels and pixel size on the [Imaging System](#) page first; without them the mosaic form is unavailable.

Note: Panels are created with no filters. Give panel 1 its exposure plan, then use **Copy this panel's exposure plan to all** on the mosaic card.

A mosaic is limited to 100 panels, and it is created all-or-nothing: if any panel name would collide with an existing target, nothing is created.

The Discover Targets button

Discover Targets... answers "what should I image?" rather than "where is this thing?" It ranks the whole catalog for your rig, site and months. See [Discover Targets](#).

The target card

Each target is a card. The collapsed header holds:

- A **drag handle** (Active column only) for setting priority.
- The **enabled checkbox** (Active cards only). Unticked, the target stays on the list but is not scheduled tonight.
- **Display name** — editable. See [Renaming a target](#).
- **Catalog ID:** — SIMBAD's identifier, shown when it differs from the display name.
- A summary line: RA and Dec in degrees, and the object type.
- A completion bar and **N% complete**. It counts accepted integration time against planned time, per filter, and a filter that has collected *more* than planned earns no extra credit — so a target with three times its Ha and no OIII does not read as done.
- Two small charts, one above the other: the **Year** chart and the **Tonight** chart (see below).
- **Expand / Collapse, Frame** (opens the Sky tab's Frame mode on this target, without expanding the card; on a mosaic it frames the whole mosaic), → **Planned** or → **Active**, **AstroBin**, **Mark reviewed** (or **Reviewed date**), **Duplicate** (see below), and ✕ to remove the target (with confirmation — removal cannot be undone).

Duplicate

Duplicate makes a second target from this one: same coordinates and rotation, same template, minimum imaging time and filter rows with the same *planned* times. It asks for no confirmation, because nothing is destroyed.

What it does **not** copy is progress. The copy's accepted counts are zero and its Complete and Reviewed marks are cleared — those describe frames on disk for the original, and carrying them would both mislead you and quietly reduce what the scheduler asks for.

The copy lands in **Planned**, never Active, whatever the original was. Two Active targets on the same patch of sky would both go into tonight's plan and the mount would shoot the same field twice, with nothing looking wrong.

Move it to Active when you mean it.

Until you compute a seasonal plan the copy sits at the **end** of the Planned column — that column is ordered by when each target is next due, and a brand-new one is not due yet.

Use it to shoot something again at different exposures, or to keep a reframe beside the original instead of overwriting it (see [Moving a target with accepted frames](#)).

On a **mosaic card**, Duplicate copies the *whole* mosaic as a new one — from whichever panel you press it on — keeping each panel's own number, so a mosaic missing Panel 3 copies as 1, 2, 4. It is refused, with nothing created, if a target already carries one of the new panel names.

The year chart

The chart beside each name shows, for the twelve months centered on today, how many hours per night the target spends above your altitude floor during darkness — your custom horizon if one is loaded, otherwise the flat **Min Altitude (°)** from the [Imaging System](#) page. The vertical line is today; faint gridlines mark months; a "10h" label anchors the scale. Every card uses the same scale and the same months, so you can read one target's season against another's straight down the column. The dates under the curve are the nearest stretch of nights when the target is up at all. The chart is captioned **Year** in its corner.

The Tonight chart

Under the year chart, a second chart captioned **Tonight** shows how the current night looks for this target, from sunset to sunrise. It is on every target card and, for a mosaic, every mosaic card (measured at the mosaic's center). Every card uses the same time axis, so you can compare targets straight down the column.

- **The target's altitude** is the solid curve in the same color as the year chart's. The scale is fixed at 0 to 90 degrees, with faint lines at 30 and 60.
- **The shaded area** under the curve is the usable dark time: the stretches when the target is above your altitude floor during darkness. Its time is counted the same way as the hours on the year chart (same rule, same horizon).
- **The ground band** along the bottom is your horizon: the flat **Min Altitude (°)** from the [Imaging System](#) page, or your custom horizon if one is loaded. It follows the floor in the direction the target is in at each moment, so a tree line or a roof shows up exactly when the target is behind it.
- **The Moon** is the dashed curve. A small disk on it shows the phase, and sits where the Moon is now while it is up, or at its highest point of the night otherwise. There is no disk if the Moon does not rise during the night. Its altitude is the same figure the scheduler uses.
- **The now line** is a vertical line, drawn only while the night is in progress.
- **The faint bands** at either end are twilight, before and after the dark window.
- **The hour labels** along the bottom are local time at your site, every three hours (spaced more widely on a very long night, such as a high-latitude winter).

The chart shows the night in progress and moves on to the next one at local noon: from midnight until noon it still shows the night that just ended (with no now line, since that night is over), and from noon on it shows the coming night. At a high latitude in summer, when the Sun sets but never gets low enough for the twilight you chose, the

chart shows only twilight and nothing shaded; when the Sun does not set at all there is no chart, only its caption. Hover the chart for a reminder of the marks.

AstroBin

AstroBin opens AstroBin's own search for this object in your browser — useful for seeing what others have captured and how they framed it. It searches by SIMBAD identifier when there is one, then the mosaic name, then the display name.

Reload from disk

Reload from disk... re-reads the config file this tab has open, throwing away what is in memory for it. Use it when something else changed the file while the app was running — the NINA plugin saving on its Targets page, a Framing Assistant target, or `nightfront-cli.exe`.

It asks first, and tells you what it found and what goes:

- Whether this tab has **unsaved config changes**. There is no "modified" marker in NightFront, so this is a comparison of the whole config against what was last loaded **or saved** — it can speak for what the file holds, and nothing else.
- **Lost either way**, because they are never written to the config file: the GA's Pareto front and the plan you selected from it, plan scoring, undo history, which cards are expanded, and which panel each mosaic card is showing.
- **Plan variants come back**, from their sidecar file beside the config.
- The forecast is normally served from cache, so a reload does not usually spend credits.

The button is disabled while anything is reading or writing the target list — a nightly solve, a seasonal roadmap run, a plan generate, or target resolution after a load.

Coordinates, templates and filter plans

Expand opens the body of the card.

Coordinates and rotation

RA (hours, minutes, seconds), Dec (degrees, minutes, seconds) and **Rotation Angle (°)** (−360 to 360) are editable. Each field commits when you press Enter or click away, not on every keystroke. The rotation is east of north — the Position Angle of NINA's sequencer target, which the Framing Assistant's Rotation box also shows, and the same number NightFront exports for NINA's center-and-rotate step.

Changing any of these moves the target. If the target already has accepted frames, you will be asked first — see [Moving a target with accepted frames](#).

A card's **Frame** (in its action column, under **Expand**, so you do not have to expand the card) opens the ♦ Sky tab in Frame mode for this config: your camera's field at the target's rotation, drawn exactly as NINA's Framing Assistant

draws it. On a mosaic panel it shows the whole mosaic with that panel highlighted. Move or turn the frame there and press **Move and save**. A target on the Completed strip has its own **Frame**, which is view only. If you have moved or turned the frame and then open another one (another card's **Frame**, a click on one of the dashed targets, or Discovery), NightFront first asks **Discard this frame and open ...?**; **Keep** leaves your frame as it is.

Templates and minimum imaging time

Load template: Select... lists the exposure templates from the [Imaging System](#) page (the row appears only once you have at least one template). Choosing one **replaces every filter row** on the target with the template's filters, with Accepted set to 0.

Min imaging time: 15m, 30m, 1h or 2h (default **1h** for a new target). Once a planner starts imaging this target it stays for at least this long, and a target is only started when it has at least this much uninterrupted time available.

The filter table

One row per filter:

Column	Default	What it means
Filter	L	Chosen from your filter registry
Planned	36	Total frames you want at this exposure
Accepted	0	Frames you have kept after review
Remaining	—	Planned minus Accepted; this is what the planners try to schedule
Plan (h)	—	Planned integration in hours (read-only)
Acc (h)	—	Accepted integration in hours (read-only)
Rem (h)	—	Remaining integration in hours
Exp(s)	300	Sub length in seconds
Gain	0	Camera gain
Offset	10	Camera offset
Readout	<i>default</i>	Readout mode number; blank means the camera's own default, not mode 0
Dither	1	Dither after every N frames of this filter
Moon Profile	— none —	The moon-avoidance profile for this filter

✕ at the end of a row removes that filter. + **Add Filter** adds a row, pre-filled with the moon profile set as the default for that filter name, if any. Below the last filter row, a read-only **Total** row sums **Plan (h)**, **Acc (h)** and **Rem (h)** across every filter on the card (a mosaic panel's total covers that panel only).

Heads up: NightFront stores integration *time*, and frame counts are a view of it at the row's exposure. Changing **Exp(s)** keeps the hours and rescales the frame counts: 120×600 s becomes 240×300 s, still 20 hours. The Planned and Accepted numbers visibly jumping is the sign your edit landed. **Plan (h)** and **Acc (h)** are computed straight from those stored seconds, the same way **Rem (h)** always has been — never from frame count $\times$ Exp(s), which would be wrong for a row shot partly under a different exposure.

A filter with no moon profile falls back to a single flat minimum moon separation — 30° unless the config file says otherwise; there is no on-screen control for it. Moon profiles themselves are defined on the [Imaging System](#) page.

Mosaic cards

A mosaic shows as **one card**, not one per panel.

The header carries a **Mosaic name** field, a "MOSAIC · N panels" line, the mosaic's center, a completion bar with accepted and planned hours across all panels, and one year chart at the center. The header checkbox, → **Planned** / → **Active**, **Mark reviewed** and X all act on **every panel**. The checkbox shows a dash when only some panels are enabled — worth fixing, because a partly-enabled mosaic plans a partial frame without any warning.

Expanded, the card has two parts.

Settings for the whole mosaic:

- **Min imaging time (all panels):** with a note of how much dark time one full round takes.
- **+ Add panel** — adds a panel copied from the selected one (pointing and filter plan, Accepted zeroed). Move it into place afterwards.
- **Copy this panel's exposure plan to all** — after confirmation, replaces every other panel's filter rows with this panel's. Each panel keeps its own Accepted counts.

The selected panel: a tab per panel (a dropdown above eight panels), a **panel enabled** checkbox for that panel alone, **Remove panel**, and the same coordinates and filter table as an ordinary target.

- Removing a panel does **not** renumber the rest. Removing panel 3 of 4 leaves Panel 1, 2 and 4, because a panel's name is how its collected frames are tracked.
- If some of a mosaic's panels ended up in the other list, the card says so. Its controls still act on all panels.

Renaming a mosaic renames every panel, so it asks first: the plugin's count of frames collected but not yet reviewed is kept by name, and those frames will be shot again. A name that would collide with another target is refused.

Mosaics laid out mirrored against NINA

NightFront lays new mosaics out exactly as NINA's own Framing Assistant does. A mosaic created by an older NightFront, or by an AstroPM import, may instead be the **mirror image** of the grid NINA draws at the same rotation. Each panel is still imaged at its own coordinates, but NINA turns the camera to the stored rotation while the panels were tiled for its mirror, so every frame sits askew of its neighbors: the overlaps shrink and gaps can open between panels. This happens only at a rotation that is not a multiple of 90°, and it is worst at 45°, 135°, 225° and 315°, where each frame is turned a full 90° against its tiling.

When the open config has such a mosaic that you have not already decided about, a red banner above the target lists says "*1 mosaic is laid out mirrored against NINA: its frames will not tile as planned.*" with a **Review...** button. Once every listed mosaic is dismissed or has a completed panel, a quieter line says how many are dismissed or completed, with **Review...** beside it. Neither appears until the target list has finished loading, or for a config that has never been saved.

Review... opens a dialog with a card for each affected mosaic: *"Stored at 160.0°; NINA draws these panels at 200.0°"*, then the panel count, the accepted hours so far, and whether it is Active or Planned, disabled, partly complete or dismissed. A mosaic with accepted frames also says it is partly shot, because either correction leaves those frames on a different layout from the frames still to come. Nothing is preselected and nothing is written until you choose, per mosaic:

- **(a) Turn the camera to *angle*...** — keeps every panel exactly where it is and rewrites the stored rotation to the angle NINA draws that same grid at (of the two angles that give the same grid, the one the rotator turns to least). Because a mosaic's overlap only tiles correctly at the rotation it was framed at, **this resets every panel's Accepted count to zero and you will need new calibration flats at the new rotator angle.** Frames shot at the old angle do not tile the corrected grid, so keeping their counts would overstate progress.
- **(b) Move the panels...** — keeps the stored rotation and moves the panels onto NINA's layout at it, each keeping its column and row in the grid and its own exposure plan. A panel that does not really move (within 5" of where it was) keeps its Accepted count; the rest reset the same way a real move always does — see [Moving a target with accepted frames](#).
- **(c) Leave it and dismiss** — the mosaic stays as it is (frame it one panel at a time in NINA), and the banner stops counting it. **Un-dismiss** on its card flags it again at any time.

(a) and (b) always show a confirmation first, even when nothing has been accepted. It says how many hours of accepted counts are reset and, after (a), the angle to take new flats at; the button reads **Correct and reset** when counts are lost, **Correct** otherwise. Every choice, a dismissal included, is written to the config file straight away, the whole mosaic in one save or not at all — there is nothing to Save afterwards.

A mosaic with a completed panel is listed **read-only**: it can be neither corrected nor dismissed until you use **Plan again** on it in the [Completed](#) strip, because a mosaic is corrected whole or not at all. If the config changed on disk since you opened it (for example, the NINA plugin moved a target), the choice is refused and nothing moves; use **Reload from disk**, then review the mosaic again.

A mosaic at 45°, 135°, 225° or 315° needs the camera's size to be checked. At those angles an older mirrored layout and NINA's own fit the same panels, with rows and columns swapped, and only the camera's shape tells them apart. If the [Imaging System](#) page states no **Sensor width (px)** and **Sensor height (px)**, a note under the banner names each such mosaic and asks for them. A mosaic that is a single row or column is told apart by the length of its step against the camera's field of view instead, so it also needs **Focal length (mm)** and **Pixel size (μm)**. Once they are entered, a mirrored one appears in the Review list like any other, and one laid out as NINA does disappears from the note. A mosaic you have dismissed, or one with a completed panel, is not named in the note. A mosaic within 0.1° of a right angle is never flagged: its frames sit at most 0.2° askew.

Note: A mirrored single *column* cannot be told from one laid out as NINA does, because its step is also a valid NINA step at a larger overlap. It is not flagged, and it frames without gaps.

An older mosaic on a very wide field at 45°, 135°, 225° or 315° is named, not corrected. Older NightFront versions sized a frame slightly narrower than NINA does, which matters only on a field wider than about 23° (a camera lens). At those four angles such a mosaic no longer matches the camera's shape either way round, so NightFront cannot tell whether it was laid out mirrored. A note under the banner names it and says it was laid out by an older NightFront; it is not offered in **Review...** Frame its panels one at a time, or create the mosaic again with the same center, rotation and grid, which lays it out exactly as NINA does.

What the NINA plugin's whole-mosaic Frame says about these mosaics. It uses the same rule as this page, so the two never disagree:

- A mosaic this page flags is refused for whole-mosaic Frame, naming an older NightFront or an AstroPM import and pointing you here to correct it. Its panels still frame one at a time.
- A mosaic at 45°, 135°, 225° or 315° whose camera is unknown is refused with a request to enter the sensor size (and, for a single row or column, focal length and pixel size) on the Imaging System page. The plugin sends NINA's connected camera for this, but the camera stated here wins whenever it is stated. If NINA's camera shows such a mosaic is mirrored while this imaging system states no sensor size, the refusal asks you to enter the size here first: until then this page names the mosaic in its note rather than offering **Review...** for it.
- An older mosaic on a very wide field at one of those four angles is refused, saying it was laid out by an older NightFront; frame its panels one at a time.
- A mosaic within 0.1° of a right angle is not flagged here, but if it is too wide for NINA's own grid to land on every panel, whole-mosaic Frame refuses and points you to framing its panels one at a time. There is nothing to correct.

Renaming a target

The **Display name** field commits when you press Enter or click away — not while you type.

Every target also carries a permanent, hidden identity that a rename does not change. The NINA plugin's count of collected-but-not-yet-reviewed frames follows that identity, so renaming a target no longer makes those frames get shot again.

A rename is refused, with the reason under the name field and the old name put back, when the new name is already another target's (or a mosaic's) or can't be a folder name: a character such as `:` or `/` is named, as are reserved names like `Completed`. Once you save, a managed rig's project folder follows the rename (see [The managed layout](#)). Renaming a mosaic also renames its panel folders under `Originals`, `Calibrated` and `Stacked`, the stacked masters' records follow them, and frames still carrying the old panel name (even ones not yet organized) are filed and counted under the new one. A carry that can't be completed is reported in the **Folder changes** note.

Heads up: That protection covers frames the plugin counted from this version onward, with a plugin new enough to read the identity. Frames counted before then are still matched by name, so renaming loses them. If in doubt, review (or **Mark reviewed**) before renaming — that settles the count either way.

Heads up: The name is still the target's identity in the exported plan, in your file and folder names, and in any automation keyed on target name. A rename changes all of those.

A rename carries the target's priority rank with it.

Priority order

The **Active** column can carry a priority order that the non-GA planners follow. The caption on the **Active** heading's own row (which keeps the Active and Planned columns starting level) tells you which order you are looking at:

- **Plan order — drag a target to set a priority instead** — no priority stated yet.
- **Priority order — highest first** — you have dragged at least once.

Drag a card by its handle to move it. Your first drag keeps everything else exactly where it is on screen and starts a priority order from that. A mosaic drags as one unit and its panels share one rank. **Reset to plan order**, beside the caption on that same row, clears the priority order entirely.

Who reads it:

- **Greedy** fills the night in priority order.
- **Proportional** uses it to decide who goes first among targets behind their share.
- **Dispatch** scores it through its **Priority order** rule.
- The **genetic optimizer** does not plan by priority, but it starts each run with one Greedy plan in its population, and that seed follows your order.

The Plan tab's Gantt rows and Target Summary are drawn in this order too. See [Planning a night](#).

Ranks survive a trip to **Planned** and back. Targets you have never ranked sort after the ranked ones. The **Planned** column has no priority order.

Reviewed stamps

Mark reviewed stamps the target with today's date and the button then reads **Reviewed** *date*. Click it again any time to restamp.

The NINA plugin counts the frames it shoots so an unattended backlog is not re-shot. A reviewed stamp tells it you have looked at everything collected up to now, so it clears its own count for that target and relies on your

Accepted numbers instead.

- Stamp after you have reviewed a target's subs, **including when you rejected every one** — Accepted does not change in that case, and the stamp is the only way the plugin learns those frames were dealt with.
- On a mosaic, the header stamp covers every panel.
- Like other edits, the stamp reaches the plugin only once you **Save** the config.

Moving a target with accepted frames

Frames collected at one pointing do not belong to a different composition. So whenever a target really moves — by typing new RA, Dec or rotation, or by **Move and save** on the Sky map — NightFront applies one rule:

- **Tiny changes are ignored.** A shift under 5 arcseconds, with rotation within 0.1° , changes nothing; a line at the foot of the page says so.
- **Small turns on the Sky map.** In Frame mode a turn is stored to 0.01° . Accepted frames are kept while the target stays within 5 arcseconds, and turned less than 0.1° in all, of where it was when its accepted counts were last reset; the move that goes past that asks first, as below.
- **Nothing accepted yet** — the move applies immediately.
- **Accepted frames exist** — a **Move ...?** dialog shows how far the target moves, the rotation change (old → new), and the accepted hours per filter. **Move and reset:**
 - sets every filter's **Accepted** to zero,
 - stamps the target **Reviewed** as of now,
 - leaves **Planned** unchanged, so its remaining time goes back to the full plan,
 - resets only this panel on a mosaic; sibling panels keep their counts.

Cancel, and the fields go back to the stored values.

A line at the foot of the page records what the move did, including when counts were reset.

Heads up: A confirmed move (and one that needed no confirmation) is **saved to the config file immediately**, unlike ordinary edits. If the config has never been saved, the note tells you to save it or lose the change.

Heads up: The frames themselves stay in your Projects tree. The next **Accept lights** on the [Processing](#) tab will offer to count them again. To make the zero stick, move this target's old lights into the rig's `rejected` folder.

If you were correcting a typo in a coordinate rather than recomposing, confirm the move and then type the Accepted counts back into the filter table.

A completed target cannot be moved; clear its completion first.

Moving a target from NINA instead

The NINA plugin's Options → Targets page can do the same thing from NINA's Framing Assistant, so a recompose at the mount does not mean coming back to this app. Frame the field, press **Preview**, pick the target from the list beside Create, and press **Update**.

The rule is the same one described above — every panel that really moves has its Accepted counts zeroed and is stamped Reviewed, and panels that did not really move are left alone. There is no dialog: the line above the button says what it is about to do before you press it, and the plugin refuses outright rather than asking when the panel counts do not match, the target is complete, or the framing is more than 5° from where that target is.

It edits the same config file this tab has open. If the app is running, use **Reload from disk** afterwards to see the new coordinates and the reset counts.

The project line and jumping to Processing

Every card carries a **project line** in the same place: the target's project folder in the [Processing](#) tree (with its status folder in a managed rig), how many lights it holds and how many are waiting to be calibrated. In place of that it can say that the folder is somewhere NightFront leaves alone, needs your answer, or doesn't exist yet. While a rename or a move between Active and Planned is not saved, the line reads "rename pending — save to apply" (or "move pending") and Save is highlighted; once you save, the project folder follows the target (see [The managed layout](#)).

Open in Processing › on the line (greyed when there's no folder yet) switches to Processing › Projects, selects the folder's status chip if it isn't showing, scrolls to the project and rings it for a moment. The project card's target name links back here.

Completed targets

Projects are marked complete from the [Processing](#) tab (**Complete**). A completed target disappears from both **Active** and **Planned** and is not planned anywhere, but it is kept in the config.

The **Completed (N)** strip below the lists expands to show each completed project (one row per mosaic) with the date it was completed. Each row has **Open in Processing** ›, which switches to Processing › Projects with the **Completed** chip selected and that project's row briefly highlighted; it is greyed when no completed project folder is known for the target (a folder with no stored link). **Plan again** clears the completion for the whole group; the target returns to the list it was in and **Save** highlights, because the change is not on disk until you save. In a managed rig the project folder also comes back out of `Completed\` after that save (into Active or Planned, listed in the **Folder changes** note); it does not bring back folders or files that completing the project removed.

The Recommender

Recommender suggests which **Planned** targets to move to Active, to fill the open time. Chosen at the top of the Targets page (beside **Targets**), it looks for the parts of each night that nothing Active can image and suggests which **Planned** targets would fill them. It ignores weather entirely; the only sky it assumes is this site's usual clear-sky fraction, the same one the Seasonal Plan tab uses.

Press **Analyze**. It always looks at the next 30 nights, starting from the planned night. That is one full lunar cycle, so you see the effect of moon avoidance on every phase of the Moon; it is not a limit on which targets it can suggest. It runs only when you press it. When you change anything that affects planning afterward, the result is marked stale rather than quietly updated. The first night is shown as a whole night, not clipped to what the forecast covers.

The chart has one row per night, from the earliest dusk to the latest dawn. Each stretch of the dark hours is one of these:

- **Covered**: at least one enabled Active target can be imaged.
- **A gap**, colored by cause: every Active target is *below the horizon* (or your custom horizon); every one is *blocked by the Moon*; or a mix, where neither cause applies to every target. When no Active target has work left, the whole night is a gap.
- **Projected idle**: some Active target could image, but its work is projected to be finished by then.

- **Too short to start:** a gap at the end of the night with no room to slew to a new target.

Gap minutes are counted in whole time bins. A night's last partial bin counts in full, as on the Feasibility panel, so a figure can read about 2 minutes high. Only the start-before-dawn limit is modeled; a short gap in mid-night where a slew would not fit is an accepted approximation.

Projected idle time is a projection and says so. Each night's time is shared among the Active targets that are up (by your priority order when you have stated one, equally when not), and each target's remaining work shrinks at this site's expected clear fraction for that night's month times the expected time-on-sensor fraction. When the site has no climate data the note reads *using the default 50 % clear*.

The suggestion. NightFront looks for the Planned target, or combination of targets, that fills all the open time: the gaps and the projected idle time together. It picks Planned targets one at a time, each the one that fills the most of what is still open given those already picked, and stops when the open time is filled or no other target adds anything. One sentence states the result: the suggested targets, how much of the gap time they would fill ("% of the gap time"), how much of the projected idle time ("X of the Y of projected idle time"), and how much stays open. Under it, each stretch still open is listed with its nights, or with the reason no Planned target can fill it. Clicking a name in the sentence scrolls to that target's row and briefly highlights it.

Each open stretch is named by where in the night it falls, from the whole stretch it covers: *Evening* when it ends before about the middle of the night, *Morning* or *Before dawn* when it starts after, *Evening to dawn* or *All night* when it runs on until dawn, and *Mid-night* when it crosses the middle of the night without reaching either end.

The list under the chart is short: the suggested targets first, marked **suggested**, in the order they were picked, then five alternatives, ordered by how much each would add beside the suggested set. **Show all** under the list (with how many more there are) adds the rest in the same order; **Show fewer** puts them away again. The column **Imaging time (next 30 nights)** shows a suggested target's own total over the 30 nights and the nights it falls on (for example "106 h over 20 nights"), and an alternative's as **Adds** that much beside the suggested set. Each row also shows when in the night it fills and the best airmass there. Hovering over a row (or clicking it) outlines where that target would fill on the chart. A mosaic is one row. A Planned target with no filters is listed as one that cannot be evaluated.

Ties. Rows that fill the same time to the minute are marked **tied** (hover over the mark to see the rule), and a note under the list says how a tie is settled: by the target that fills more of the gaps first, then by your priority order, then by the target that still needs the most integration, then by name.

The gap types stay on the chart as colors, explained in the legend above it; clicking a stretch on the chart highlights it and fades the others.

Under the list, side notes (never recommendations) list an Active target you have disabled that would cover some of the gaps, an Active target with no feasible time in the horizon, and each Active target's remaining work against its projected finish.

Logs line. Under the suggestion, one line compares the projection with what your NINA logs show. Once you have run **Analyze** on the Analysis page (or chosen log files there), it reads "The N imaging nights analyzed ended M min before dawn on average." ("The imaging night analyzed ended M min before dawn." for one night). It counts only the nights you selected there that have at least one light frame; a night lost to weather is left out. With no completed analysis the line points you to the Analysis page, and when the analysis holds no imaging night it says

"The logs analyzed show no night to compare with." The line is for your judgment only; it never changes the suggestion or the ranking.

→ **Active** is the same button as on the Targets page: it moves the target (or the whole mosaic) as an unsaved edit and highlights **Save**. From here it also analyzes again at once, so the next open gap rises to the top; the old result stays on screen marked as updating until the new one arrives. The button is disabled once a row is Active and while an analysis runs. NightFront never moves a target on its own.

Jumping to a target from elsewhere in the app (for example from Processing) switches back to the **Targets** view.

Saving target changes

Save writes the open config; **Save As...** writes it somewhere new. Edits on this page — filters, counts, enabled, category, names, reviewed stamps, priority — stay in memory until you save, and the NINA plugin only sees what is on disk. The one exception is a target move, which saves for you.

Discover Targets

Discover Targets..., the fourth button on the [Targets](#) page, opens the **Target Discovery** window. It ranks thousands of catalogued objects for *this* rig, from *this* site, during the months you choose, and lets you add the ones you like as Planned targets. The same window can frame any candidate on the Sky map at the grid the search chose.

Target Discovery — CDK14

Months

Jan

Feb

Mar

Apr

May

Jun

Jul

Aug

Sep

Oct

Nov

Dec

All

None

Filters

all types, all catalogs

Accept up to

1 frame

2 panels

4 panels

9 panels

Search

Refresh catalogs

Show results on Sky map

Field 0.80° × 0.54° · f/7.2

22261 objects · downloaded 2026-09-14

Ranked 900 of 22261 objects in 2.1 s · 18403 ever rise, 17478 get enough time

Narrow the results by name

Object	Type	Score ▾	Hours	Fit	Grid	Reach	Size	Mag	Flags	In frame			
M 3- 4	open cluster	0.704	590 h	1.00	1	1.00	23'	5.2		—	Frame	AstroBin	+ Planned
h Persei Cluster	open cluster	0.703	620 h	1.00	1	1.00	14'	3.7		Double Cluster	Frame	AstroBin	+ Planned
NGC 1245	open cluster	0.665	602 h	0.96	1	0.97	11'	8.4		—	Frame	AstroBin	+ Planned
NGC 744	open cluster	0.662	608 h	0.96	1	0.98	12'	7.9		—	Frame	AstroBin	+ Planned
chi Persei Cluster	open cluster	0.662	621 h	0.94	1	1.00	11'	3.8		Double Cluster	Frame	AstroBin	+ Planned
NGC 957	open cluster	0.650	626 h	0.93	1	0.99	10'	7.6		—	Frame	AstroBin	+ Planned
NGC 1502	open cluster	0.642	636 h	0.93	1	0.99	10'	6.9		—	Frame	AstroBin	+ Planned
NGC 1664	open cluster	0.635	563 h	0.96	1	0.99	11'	7.6		—	Frame	AstroBin	+ Planned
NGC 1528	open cluster	0.633	600 h	0.91	1	1.00	10'	6.4		—	Frame	AstroBin	+ Planned

Flags mark a score resting on an assumption: size? — no size catalogued · shape? — only one axis known for an object that is usually elongated · sb~ — surface brightness derived from magnitude and size, not stated. A ~ after Reach means the catalog holds no photometry for that object, so a flat value was used.

Catalog data: OpenNGC by Mattia Verga (CC BY-SA 4.0, github.com/mattiaverga/OpenNGC) — the merged local copy is an adaptation under the same license. Sharpless, Lynds, Barnard, van den Bergh, Green, Strasbourg-ESO, Abell (ACO 1989), Arp (Webb 1996), Hickson and Harris catalogs via the Vizier service, CDS Strasbourg.

The catalog store

The first time you open the window there is nothing to search: **No catalog downloaded yet**. Press **Fetch catalogs**. NightFront downloads:

- **NGC, IC and Messier** from OpenNGC, and
- **Sharpless, Lynds Bright and Dark Nebulae, Barnard, van den Bergh, Green's supernova remnants, the Strasbourg-ESO planetary nebulae, Abell galaxy clusters, Arp peculiar galaxies, Hickson compact groups and Harris globulars** from the Vizier service.

The sources are merged into one row per physical object carrying every name it is known by, so NGC 7000 is also Sharpless 2-117 and Caldwell 20. A progress line shows each source as it arrives.

- **It runs once.** The merged catalog is saved in `~/.config/nightfront/catalog-cache.json`, and from then on searching works offline. The status line shows the object count and the download date.
- **Refresh is up to you. Refresh catalogs** downloads everything again; nothing refreshes on its own schedule. If a refresh fails, the catalog you already had stays in place.
- If the download succeeds but cannot be saved, the window says so — otherwise it would quietly download again next time.

The footer credits the data: OpenNGC by Mattia Verga (CC BY-SA 4.0), with the other catalogs via Vizier at CDS Strasbourg. The merged local copy is an adaptation under the same license.

Choosing what to search

Choosing months

Months has a chip for each month; the current month is in bold. The window opens with this month and the next two selected. **All** and **None** select everything or nothing. Pick any combination, including months that are not next to each other — October and March is a fair question.

- A month means its **next** occurrence. Choosing October in January means this coming October.
- Choosing the current month counts its remaining nights plus the same month next year.
- **Search** stays disabled with no month selected.

The months change the *ranking*, not just the list: an object's available hours are totaled over the nights you asked about.

Type and catalog filters

► **Filters** expands the filter chips; the collapsed line summarizes what is selected ("all types, all catalogs" when nothing is). **Clear** removes every filter.

Row	Chips
Catalog	Messier, NGC, IC, Caldwell, Sharpless, LBN, LDN, Barnard, van den Bergh, SNR (Green), PN (SEC), Abell (clusters + PNe), Arp, Hickson, Palomar, UGC
Galaxies	All galaxies, Spiral, Barred, Elliptical, Lenticular, Irregular, Edge-on, Face-on, Pairs, Triplets, Groups, Clusters, Peculiar (Arp/VV)
Nebulae	All nebulae, HII / emission, Reflection, Dark, Supernova remnants, Planetary
Stars	Open clusters, Globulars, Cluster + nebula, Asterisms / star clouds

- Chips add together: selecting **Spiral** and **Planetary** shows both. No chips selected means everything.
- An object is "in" a catalog if any of its names is from that catalog.
- **Edge-on** is a galaxy whose minor axis is under a quarter of its major axis; **Face-on** is over 80 %. **Abell** covers both Abell's galaxy clusters and his planetary nebulae, which share the name.
- Filters decide which objects compete. They never change how an object scores.
- Filters take effect on the next **Search**.

A line under the chips lists classifications the catalogs cannot support (flocculent and grand-design spirals, rings, polar rings, shells, tidal streams, chains, Wolf-Rayet, YSOs, planetary-nebula morphology), so there are no chips for them.

Mosaic size and Search

Accept up to — 1 frame, 2 panels, 4 panels or 9 panels — is the largest mosaic you are willing to shoot. For each object NightFront tries every grid up to that size and keeps the one with the best score, after charging for the extra panels.

Search runs the ranking. It needs:

- **Your rig's field of view.** Without focal length, sensor size in pixels and pixel size on the [Imaging System](#) page, the window refuses to search rather than rank on a different question. When optics are set, a line shows the field and focal ratio.
- A catalog loaded and at least one month selected.

Your site location and twilight setting, and the **Min Altitude (°)** and custom horizon from the [Imaging System](#) page, all come from the open config.

Reading the results

A summary line reports how many objects were ranked, how long it took, how many ever rise, and how many get enough time. Two notes may appear under it:

- **No climate data:** "Weather is not in this ranking..." — without your site's climate profile, objects in your cloudiest months rank as if those months were as clear as the rest. Fetch the profile on the [Site](#) page (or enter monthly values by hand).
- **No focal ratio:** Reach is on a nominal f/4 scale. The order is unaffected.

Narrow the results by name filters the list by any designation or common name without searching again.

Note: An object needs at least **10 expected clear dark hours** above your floor over the chosen months to be listed at all. Below that it is left out, not ranked low — no plan could make real progress on it.

Result columns

Click **Score**, **Hours**, **Fit**, **Reach**, **Size**, **Mag** or **In frame** to sort by that column (a ▼ marks the active sort). Hover a header for a short explanation.

Column	What it tells you
Object	The object's name
Type	The catalog's classification (galaxy, emission neb, open cluster, ...)
Score	The overall ranking: availability × framing fit × reach, discounted for the number of panels. It is a product, so an object must be decent on every count — excellence at one thing cannot rescue a poor showing on another.
Hours	Expected clear dark hours above your horizon over the chosen months, with your site's monthly cloud climatology applied. The most direct number here.
Fit	How well it fills the frame, 0–1. Best when the object spans about half the frame's limiting dimension; smaller wastes field, larger clips the object.
Grid	The mosaic that fit was computed for: 1 for a single frame, otherwise columns × rows
Reach	How reachable it is with this rig, 0–1: surface brightness against your focal ratio, or integrated magnitude for star clusters. A narrowband or OSC filter in your filter registry helps emission objects.
Size	Major axis in arcminutes
Mag	Magnitude — V where the catalog has it, otherwise B. For an Abell galaxy cluster it is the tenth-brightest member's, not the cluster's total.
Flags	Where the score rests on an assumption (below)
In frame	Other catalogued objects whose centers fall inside the same frame or grid

A "—" in **Size** or **Mag** means the catalog does not state one; those objects sort last.

Note: Reach measures whether you *can* get an object, not whether it is worth getting. On a long focal length the top of the list can be dominated by bright open clusters. Use the type chips to steer toward what you want.

The score's availability term weights hours by airmass, so time high in the sky counts for more than time near the horizon; **Hours** is the plain total.

Flags and the Reach tilde

The **Flags** column marks rows whose score leans on an assumption. The legend under the table repeats this:

- **size?** — no size catalogued, so framing uses a flat, deliberately modest value.
- **shape?** — only one axis is known for an object that is usually elongated, so it was treated as round. Clusters, globulars and supernova remnants are not flagged: one diameter is all they need.
- **sb~** — surface brightness was worked out from magnitude and size rather than stated.

A ~ after a **Reach** number means the catalog holds no photometry for that object, so a flat nominal value was used. Whole catalogs — Sharpless, Lynds, Barnard, van den Bergh, Green — carry none, which is why this rides on the number rather than filling the Flags column.

Companions in frame

In frame lists up to three neighbors that share the frame at this pointing, then "(+N)" for the rest (at most eight are counted, which also caps sorting by this column). It is information only — a crowded field does not score higher. The frame used is the bounding box of the grid, unrotated, so it errs on the side of including a neighbor near the edge.

Discovery row actions

Each row ends with:

- **Frame** — opens the ♦ Sky tab in Frame mode for this config, on the object, laid out as the row's **Grid** at NINA's default 20 % overlap and rotation 0; this window stays open. **Create planned target...** there adds exactly that frame, as **+ Planned** would.
- **AstroBin** — opens AstroBin's search for the object in your browser, to see what others have captured and how they framed it.
- **+ Planned** — adds the object to your target list. It reads **on list** instead when the object is already there.

Adding a candidate

- **It lands in Planned, never Active.** A suggestion goes onto your roadmap; it cannot change tonight's plan until you move it to **Active** yourself.
- **It has no exposure plan.** Open it on the Targets page and use **Load template:** or **+ Add Filter** — until then no planner will schedule it. The note after adding says so.
- **A multi-panel row becomes a mosaic** of exactly the grid in its **Grid** column — the grid the score was computed for — with 20 % overlap and rotation 0°. "Set the exposure plan on panel 1, then copy it to all."
- Minimum imaging time starts at 1 hour for a single frame and 30 minutes per panel for a mosaic.
- Coordinates come straight from the catalog; SIMBAD is not consulted, and no catalog position angle is applied. Set rotation on the target card when you have decided on one.

Already on your list means any existing target whose name matches any of the object's designations, whose mosaic name matches, or which sits within one arcminute of it — so a target you added under your own name is still recognized. A name that collides with an existing target is refused with a message rather than added twice.

The Sky gallery

The **Sky** tab, pinned at the left of the tab strip, is a map of the sky with *your finished images* placed on it. It belongs to the window, not to a configuration: images from any rig can be posted to it, and it never writes into a config file or a Projects folder. It also works without a gallery: until you choose one, it still draws the survey and star layers.

Setting up the gallery

The first visit asks for a **gallery folder**: **Create a gallery...** for a new one or **Open an existing gallery...** for one you made before. It can be any folder, local or on a share, and it is yours alone (one writer). NightFront writes only inside it: an index (`gallery.json`), an `images` folder holding one display copy per image, and `thumbs`. **App settings > Paths** shows the folder and lets you choose another; the one open now is closed and nothing in it is deleted.

Your **original** files are linked, never copied and never changed. Removing an image from the gallery deletes only the gallery's copy and thumbnail.

The map

Drag to pan, scroll to zoom about the cursor, and use **Find...** (type a name and press Enter) to center on an object by name. The view is always north-up. A pan stops at a celestial pole, and dragging sideways while on the pole turns the view about it.

- **Survey picker.** The background is a sky survey, fetched as tiles and cached on this computer. The default is **DSS2 color** for broadband; **NSNS** (the Northern Sky Narrowband Survey) is the narrowband choice. Where a survey has no coverage, the next one shows through instead of black. If your computer has a copy of NINA's Framing Assistant cache, it is offered under "On this computer".
- **Browse more surveys...** lists the CDS registry so you can add a survey by name, ID, URL or a local folder. A survey you add is your choice; NightFront shows its own credit and terms when you add it.
- **Layers.** The **Layers** menu switches survey imagery, stars, constellation lines, constellation labels (each constellation's full name at the center of its figure, thinned out where names would overlap), constellation boundaries and an RA/Dec grid on and off, and **Image labels** hides the name and rig written under each of your images; a layer with a non-commercial license is marked **NC**. The line along the foot of the map credits every layer that is drawn, including the survey shown through a gap; longer acknowledgments (2MASS, Pan-STARRS, WHAM for NSNS) open from it.
- **Map | Grid.** The Grid view shows your images as thumbnails with the same filters and a sort menu. The **rig chips** choose which rigs' images show: click a chip to add that rig or take it out, and **All rigs** shows every rig again. **Needs placing** lists images not on the map yet. Once every image (under the rig chip) is placed, **Needs placing** turns itself off and is greyed out, so it never hides a gallery that is all placed; if filters still hide every image, the Grid says which ones are on and how to clear them. Overlapping images: the most recently captured is on top (the posted date stands in when no capture date is given). Click an overlap and that image is selected and drawn on top; click again for the next one down, and the card says how many are

under the click. A click opens the image's card; a double-click shows it on the map, and on the map itself a double-click on an image zooms to it, the same view as **Show on map**. With a card already open, a double-click on that image zooms to it even where it overlaps others.

Each image's outline says how it was placed: plate solved, from the file's header, by hand, with star pins, or snapped to the survey.

Posting images

Post images... takes JPEG, PNG or TIFF finals (8- or 16-bit, or 32-bit float; classic TIFF or BigTIFF). Drop files in or choose them (**Choose files...** opens in the folder you last chose from, or in the project's folder when you came from a Completed card); each gets a row. NightFront fills in every hint it can find in the file, or in a FITS, XISF or `.wcs` (a plate-solution file from ASTAP or astrometry.net) beside it with the same pixel size (a plate solution, RA/Dec, focal length, pixel size); fields read from a file have a dashed border and you may overwrite any of them. A `.wcs` places the image by itself only when it says which program wrote it (the two count pixel rows from opposite ends); one that does not still gives its position and image scale as hints, and ASTAP solves the image. When two rows have the same file name in different formats (a TIFF and a PNG of one picture, say) and not different pixel sizes, a line under each says it looks like the same image; it only warns, so post one and **Remove** the other (a TIFF reads faster than a large PNG).

- **Image scale** is how much sky one pixel covers, in arcseconds per pixel ("/px). If you do not know it, enter your **focal length** in millimeters and your camera's **pixel size** in micrometers and NightFront works it out: $206.265 \times \text{pixel size} \div \text{focal length}$. Multiply the pixel size by any binning, and divide the scale by any drizzle factor. Picking one of your NightFront imaging systems fills the scale from its optics.
- **Object** is a name SIMBAD knows (M 31, IC 1396). It is looked up once, now, and the coordinates are stored; leave it blank to use a position in a FITS, XISF or `.wcs` file beside the image.
- **Rig** is a free-text label for the filter chips.
- **Solve online if ASTAP can't** is off by default. See [Placing an image](#).

Post makes the display copy (longest side up to 8000 px, JPEG) and queues the placement; the dialog can close at once. Posting a very large TIFF is fine: the original is read in strips, never all at once.

From a completed project. A card on the Completed strip of [Targets](#) has **Post to Sky gallery...**, opening the same dialog already filled in (title, stored coordinates, rig, scale from the optics, integration time, and the newest image in the project folder, if this computer has the folder). Once posted the entry is independent of the project, and the card shows ♦ **in Sky gallery**, which jumps to the image.

Placing an image

An image shows on the map only once it has a position. NightFront tries, in this order:

1. **A position the file already carries:** its own header, or a FITS, XISF or `.wcs` file beside it.
2. **Guided placement opens** straight away, with the image drawn over the survey. Placing by hand is the main path.

3. **ASTAP**, a plate solver you install separately (set its path in **App settings > Paths**, or NightFront looks in its usual places), tries quietly in the background on a smaller copy. If it succeeds you are asked *Use it?*; if it fails nothing is said. It never changes your file.
4. **Refine** (below) snaps a rough placement to exact.
5. **astrometry.net**, a free online solver, only if you ask for it: **Solve online (slow)**... on the image's card, or the Post dialog's checkbox. It needs a free API key in **App settings > Paths**. NightFront says plainly that a 2000-pixel version of the gallery copy, never the original, is uploaded, marked private with no commercial use and no modifications, and asks every time.
6. **Needs placing**, if you close the placement unsaved. NightFront tries ASTAP again after a restart.

Guided placement

The image is drawn translucent over the survey. The **opacity** slider starts at 50 %, and **holding Space** hides the image so you can blink between the two.

- Drag the image to move it; drag a corner handle to rotate it (and to resize it when the scale is unlocked). Arrow keys nudge one pixel (Shift: ten); [and] rotate 0.1°.
- An image linked to a project opens at that target's center and rotation, at the scale you gave when you posted it (a typed "/px, or focal length and pixel size with binning and drizzle) and, if you gave none, at the rig's own scale; the line under **How it's placed** says which, and says so when the two differ by more than 3 % (drizzle, binning or resampling?). The scale starts **locked** ("unlock if the image was resized or drizzled"); the project's frame or mosaic grid is drawn as a dashed guide box. It is a reference only, since a crop does not fill it. The target's rotation gives the frame's outline but not which end is up: a rotator can reach the same angle turned half way round, and processing can turn a final, so the image may open upside down. **Refine** also tries it turned 180°.
- **Pin stars**. Click a star in your image (it snaps to the star's center), then the same star on the map (it snaps to the nearest catalog star). One pin fixes the shift, two fix shift, rotation and scale exactly, and three or more give a best fit with a residual in arcseconds. On a starless image, pin features of the nebula instead. The **Gaia stars** chip adds Gaia DR3 stars, fetched for the field, as snap targets (ESA/Gaia/DPAC, non-commercial).
- **Refine (snap to survey)**. From your rough placement it searches 2' of shift, 2° of rotation and 3 % of scale, matching Gaia stars first and, for a starless final or a failed star match, the survey under the image; it also tries the image turned 90°, 180° and 270°. A match is shown for you to **Save** or **Revert**, never applied silently. If nothing matches well enough it says *place it closer*. It needs the network to fetch stars or survey tiles. It fits a shift, rotation and scale; lens distortion can leave several arcseconds at the corners of a wide image.
- **Save placement** keeps it. **Cancel** without saving leaves the image where it was, or under Needs placing.

An image's card

Clicking an image opens its card: how it was placed, the center, size and angle, its integration time per filter and the original's state. **Open original, Edit details...** (title, object, rig and its imaging-system link, capture dates and notes), **Place...**, **Re-solve** (the background ASTAP retry), and a menu (⋮) with **Solve online (slow)**..., **Show in folder**, **Replace image...**, **Relink original...** and **Remove from gallery...**

Zooming into the original

Zoom past the gallery's copy and NightFront builds a tile pyramid from your linked original in the background (shown on the map, and on the image's card, as "Building tiles from the original"). Until a tile is ready the copy is drawn enlarged, never blank. The tiles live in the sky cache and are rebuilt when needed.

If the original is missing or has changed, the map keeps the copy and the card says *Original not found* or *Original changed since posting*. **Relink original...** accepts only a file with the same pixel dimensions, because the image's sky position depends on its pixel grid; **Rebuild the copy from it** makes the copy again from a changed original of the same size.

Framing on the map

The Sky map is where you frame a target, as NINA's Framing Assistant does; NightFront no longer drives Stellarium, and the Sky map replaces it. A target card's **Frame** button, a click on one of the dashed targets, or Discovery's **Frame** opens the **Frame** view for that config.

- **What it draws.** Your camera's field, or a mosaic grid (columns × rows, overlap in % or pixels), laid out exactly as NINA lays it out. The pane shows the center to 0.1 s of RA and 0.1" of Dec, and the rotation to 0.01°.
- **Moving it.** Drag the frame, drag its round handle to turn it, or use the arrow keys and []. Double-click empty sky to move the frame there, its rotation kept, then press **Create planned target...** (or **Move and save** when the frame is one of your targets); a double-click on a catalog object, a result or a dashed target opens it instead.
- **Frame for** lists every open config; one that cannot be framed (no camera, never saved, still loading or changed on disk) is listed with the reason. **Open a config...** opens one in place.
- **Saving. Create planned target...** adds the frame, or the mosaic, to the config's Planned targets and saves the config. **Move and save** moves an existing target to the frame; a move that would reset accepted counts asks first, listing each panel. Small turns are kept as counts while the target stays within 5" and under 0.1° of where its counts were last reset.
- **Charts.** The Year and Tonight charts for the chosen config's site update a moment after you stop dragging.
- **Catalogs.** The **Catalogs** layer draws Target Discovery's catalogs as outlines at their catalog size and angle, colored by type. Click an object for its card, with **Frame this** and **Create planned target....** In Frame mode each outline is tinted by whether it fits one frame, needs a mosaic, or is too small or too large. If the catalogs were never downloaded, the layer offers the one-time download.
- **From Discovery. Show results on Sky map** draws every ranked result as a numbered outline at its best grid; **Rank in Discovery...** in the Catalog filters drawer opens Discovery with the drawer's chips, and **Only objects in the current view** limits that search to what the map shows.

Where the imagery comes from

NightFront ships no sky imagery: tiles are fetched from the survey servers as the map needs them and cached per user. The credits, drawn on the map, are the ones each survey asks for. NSNS, the Gaia stars and the NINA-cache

Northern Sky option are non-commercial data: each is its own switch-off-able layer. NightFront is free and is never sold.

App settings > Paths holds the gallery folder, the ASTAP path, the astrometry.net key, a NINA sky cache (or *Use NINA's own*), and the sizes of the survey-tile and deep-zoom caches, each with **Clear**. The map's sky colors stay dark in both the dark and light themes.

Schedule

Schedule is the rail page where you decide what to image: which projects fit the months ahead, and what to shoot tonight. It used to be two rail pages, Seasonal Planning and Nightly Planning; they are now one page. Along the top are two labelled groups, each with a **Setup** and a **Plan**: **Seasonal** for the months ahead and **Nightly** for tonight. One of the four is selected at a time.

Group	Segment	What it is for	Read more
Seasonal	Setup	The horizon (1, 3 or 6 months) and the trade-offs for the multi-month roadmap.	Planning the season
Seasonal	Plan	Run the roadmap and read it: when each project gets its time and when it should finish.	Planning the season
Nightly	Setup	How hard to search, how many nights, what "good" means, and Tonight's Feasibility .	Planning a night
Nightly	Plan	Make plans with ► GA , ► Greedy , ► Proportional and ► Dispatch , compare them on one chart, pick one and Export Sequence... to NINA. Its Plan / Editor chips switch to the Editor , where you build or reshape tonight's plan by hand, block by block.	Planning a night

Schedule opens on **Nightly > Setup**, the first thing you check before a night.

How the segments fit together

The two Seasonal segments are **informational**: the roadmap never changes your targets, their Active or Planned category, or tonight's plan. It looks at Active and Planned targets alike, to help you decide what to promote and what to leave for later. The Nightly segments only ever consider enabled **Active** targets, because that is what the NINA plugin will image.

Some settings are shared between the two groups, and changing one changes it in both places. The twilight type is the main one.

The overheads (slews, flips, dithers, autofocus) belong to your rig, so they live on the Imaging System page's **Overheads** sub-tab, not here; Nightly > Setup keeps a one-line pointer to it. They are best set from what your rig really did. The **Analysis** page reads NINA's own logs and suggests values; each overhead field on that sub-tab shows the suggestion as a one-line hint with an **Apply** button, and **See Analysis > Overheads** > jumps to the full table. See [Analysis](#) and [Imaging System](#).

Everything you make on Nightly > **Plan**, with either of its chips, is listed in one place, **Your Plans**, and drawn on the same chart, however it was made. See [Planning a night](#) for how stale generated plans are handled.

Planning the season

Planning a night answers "what should I image tonight?". The **Seasonal** group of the [Schedule](#) page answers "over the next few months, when will each project get its time, and when will it finish?" It spreads your targets across every night of a one-, three- or six-month horizon, using your site's climate to estimate how many of those nights will actually be clear.

It has two segments: **Setup**, where you choose the horizon and the trade-offs, and **Plan**, where you run it and read the result.

Note: The seasonal plan is **informational**. It never changes your targets, their Active or Planned category, or tonight's plan. It is a roadmap to help you decide what to promote, what to add and what to leave for later.

What it considers:

- **Both Active and Planned targets** — unlike the Nightly segments, which only schedule Active ones. See [Targets](#) for the two categories.
- Every target that is **not complete**, whether or not its enabled box is ticked (that box only decides tonight's plan, so a target left out of tonight still appears in the roadmap).
- The same hard limits as every night: your altitude floor (Min Altitude, or your custom horizon) and each filter's moon avoidance profile, from the [Imaging System](#) tab.
- Your site's **climate profile** — the expected clear-sky fraction for each month, fetched from NASA POWER on the [Site](#) tab.

Heads up: Until you fetch a climate profile, every month is assumed to be 50% clear. Fetch one on the Site tab (or correct individual months there) before trusting the monthly figures.

Seasonal Setup

Control	Range	Default	What it does
Horizon	1 month, 3 months, 6 months	1 month	How far ahead to plan, starting from the planned night.
Twilight Type	Civil, Nautical, Astronomical	Astronomical	When each night starts and ends. Shared with Nightly › Setup: changing it here changes it there.
Expected Utilization Fraction	0.20 (Conservative) – 1.00 (Optimistic), in steps of 0.05	0.50	The share of a clear, dark night you expect to end up as time on sensor.
Planned Target Weight	0 (Finish Active first) – 1 (No preference)	0.30	How much a Planned target's progress counts compared with an Active one's.
Completion Priority	0 (Off) – 1 (Strong)	0.50	Favors finishing nearly-complete targets.
Effort Level	Fast, Balanced, Thorough	Balanced	How big a search to run.

Save your choices with **Save / Save As...** in the bar at the top of the window. The run itself is on Seasonal › Plan.

Expected Utilization Fraction, explained

The climate profile already accounts for cloud. This slider accounts for everything else that keeps the shutter closed on a clear night: slews, meridian flips, dithers, autofocus, and the clear nights you won't actually be imaging. Each night's imaging budget is its dark time × the month's clear fraction × this fraction. Slide toward **Conservative** if your rig doesn't run every clear night; toward **Optimistic** if it does and your overheads are small. It can't go below 0.20, which keeps "conservative" from producing an empty plan.

Planned Target Weight, explained

Lower values put more pressure on finishing Active targets first; 1.0 treats Active and Planned alike. It is **pressure, not a gate**: a Planned target is never blocked outright, and it still takes nights when no Active target is up.

Seasonal effort levels

Chip	Population	Generations
Fast	10	200
Balanced	50	100
Thorough	100	200

Seasonal Plan

Press ► **Run**. The button reads "Running..." while it works. It is unavailable while another open tab is computing a schedule ("A schedule is already being computed in another tab.") and while targets from a loaded config are still being resolved. When the run finishes, **Best score** shows the plan's overall score — higher is better, and it is useful only for comparing runs of the same setup.

Before the first run the page says "No seasonal plan yet. Configure it on Seasonal > Setup, then click Run above."

The plan is saved beside your config as soon as it is computed, and comes back the next time you open that config — the caption shows "N nights starting · last computed ". (A config that has never been saved has nowhere to keep it, so save the config first.)

Heads up: After a seasonal run, the Targets page may re-sort itself into plan order — soonest start first, then earliest finish. This always applies to the Planned list, and to the Active list whenever you haven't set a priority order there. Mosaic panels stay together.

Seasonal Monthly Summary

A table of hours per calendar month, lined up with the chart below it and scrolling with it:

Row	Meaning
Ideal	Total dark time in the month's nights (per your twilight type).
Expected	Ideal × the month's expected clear fraction from your climate profile.
Planned	Hours the plan actually gives to targets.
Available	Expected – Planned: clear time the plan did not allocate.

Planned can never use all of Expected, because each night's budget is also scaled by the **Expected Utilization Fraction**. At the default 0.50, even a fully booked month shows about half its Expected hours as Available; noticeably more than that means nights no target could use. A horizon that starts or ends mid-month shows a partial column for that month.

Seasonal Gantt chart

One column per night across the whole horizon. Long horizons scroll sideways.

- **Moon** — a band across the top, shading from dark (new moon) to bright (full), with a visible change in shading at a quarter illumination.
- **Active** and **Planned** sections — each target in its own group, sorted by when its time starts, then by when it finishes.
- Each target's label shows the total hours granted and the date range: "12.5h · Oct 3 → Nov 18", "... → beyond horizon" when it doesn't finish within the horizon, or "no allocation" when it gets no time at all.
- Each target has three bands: **Total** (time per night, brighter = more), then that same time split into **Broadband** and **Narrowband** by filter role. Every filter whose role isn't narrowband (luminance, OSC and so on) counts toward Broadband.
- A thick green line marks the night a target reaches its full desired integration. It only appears for targets that finish inside the horizon.

The legend reads **Total, Broadband, Narrowband, Completed** and "Moon: new → full (kink at quarter)".

Note: The moon band is for your eyes only. The seasonal planner does not prefer broadband near new moon or narrowband near full. The only way the moon affects which filters get time is each filter's moon avoidance profile, which rules a filter in or out for a night.

Diagnostics (at the bottom; open or closed as you last left it on either seasonal segment) shows how the plan's score improved generation by generation.

Planning a night

The **Nightly** group of the [Schedule](#) page — its **Setup** and **Plan** segments — is where tonight's plan (and the next night or two) gets made:

- **Setup** — how hard to search, how many nights, what "good" means, and a feasibility check before you spend a run.
- **Plan** — make plans, compare them on one chart, pick one, and export it to NINA. Two chips at the right end of its top row, **Plan** and **Editor**, switch between that and the **Editor**, where you build or reshape a plan by hand, block by block.

There are five ways to produce a night, and they all land in the same place:

Way	Where	What it does
► GA	Plan	The genetic algorithm. Searches thousands of candidate nights and returns a whole family of them, trading utilization against quality.
► Greedy	Plan	Fills the night with the highest-priority target that is feasible and still needs time, then the next, and so on.
► Proportional	Plan	Splits each night's darkness evenly between the targets that can use it; priority order decides who goes first.
► Dispatch	Plan	Scores every available target by weighted rules at each decision point and images the winner.
By hand	Plan › Editor chip	You place blocks on a timeline; the scheduler fills them.

Every one of them is scored on the same two axes and drawn as a point on the same chart, so you can see at a glance how a hand-built or rule-scored night stands against what the optimizer found. Whichever you select drives the Gantt chart, the tables and the export identically.

Utilization and Quality, in operator terms

These are the two numbers every plan gets. Both run from 0 to 1, higher is better, and they pull against each other: the plan that keeps the shutter open longest is rarely the one that images everything at its best.

Utilization — *how much of the night did you use, and did each target get its share?* It blends two things:

- **Global**: time imaging divided by the night's darkness. Idle time counts against you, and so does overhead (slews, flips, dithers, autofocus).
- **Per-target**: for each target, time imaging divided by the time it was actually imageable tonight (capped at what it still needs), averaged across targets. A target that finishes scores full marks. Targets that could not be imaged at all tonight are left out of the average rather than dragging it down.

The **Global vs. Per-Target Weight** slider on Setup sets the mix, and **Completion Priority** gives extra weight to targets that are close to done.

Quality — *how well was each frame taken?* It is mainly an airmass score: frames shot high in the sky score better than frames shot low. If you define filter preference rules, Quality becomes a blend of that airmass score and how often the plan used your preferred filters when the rule's condition held. The blend weight is the strongest rule's aggressiveness (at most 30%); with no rules, Quality is the airmass score alone.

Note: Altitude and the moon are **hard constraints**, not scores. A target below your **Min Altitude** (or your custom horizon, which replaces it) or too close to the moon for a filter's moon profile is simply never offered to any planner for that time and filter. They are set on the [Imaging System](#) tab. No slider here trades them away.

Nightly Setup

The page reads top to bottom. Changes are part of the config; use **Save** / **Save As...** in the bar at the top of the window.

Optimization Parameters section

Effort Level sets how big a search the GA runs:

Chip	Population	Generations	Use it when
Fast	100	200	You want an answer quickly or the machine is slow.
Balanced (default)	400	200	Everyday planning.
Thorough	400	800	You have a few minutes and want the best front.

The line under the chips shows the numbers. A config whose values match no preset reads "Custom: population ..., ... generations" — it is shown as-is, never snapped to a chip.

Advanced GA parameters (collapsed) holds **Crossover Probability** (default 0.80) and **Mutation Probability** (default 0.01, range 0–0.2). Most nights never need these; the effort chips are the usual way to trade run time for search.

Scheduling Criteria section

Horizon — **1 night**, **2 nights** or **3 nights**. The Plan tab tells you before a run how many nights it will actually plan ("Will plan across 3 nights (horizon 3, forecast covers 3)").

Heads up: Later nights don't change tonight's schedule — they are planned after it. What you gain is one exported plan file per night. The horizon is also limited by what the forecast source covers (72–82 hours, so often three nights). A config asking for more nights than the chips offer is kept as-is, with a note, until you click a chip.

Twilight Type — **Civil**, **Nautical** or **Astronomical** (default). This defines when the night starts and ends. It is shared with Seasonal › Setup: changing it on one changes it on the other.

Balance time across a mosaic's panels (off by default) — when this is on, the optimizer treats a mosaic as **one** choice and gives each turn to whichever panel is furthest behind, so its panels finish together instead of one at a time. A session with no mosaic in it is unaffected either way.

Hold a panel until a sibling falls N behind (1 hour by default) — how long a panel keeps the mount once balancing is on. The number is **hours the panel spends imaging**, so one hour means the same thing on a panel that wants ten hours as on one that wants a hundred, and on a cloudy night as on a clear one. The slider runs from zero to eight hours in quarter-hour steps, and it does nothing unless the checkbox above it is ticked.

At **zero** a panel hands the mount on the instant it stops being the one furthest behind. That keeps the panels closest together, and it is also why a balanced mosaic could not hold a long window while any ordinary target competing for the same patch of sky could — in testing that moved a two-panel mosaic's entire night to a single-pointing target 17° away.

Raising it trades that evenness back for longer uninterrupted visits and fewer slews. Measured over three nights at a threshold of roughly this size: about **half the slews**, visits roughly **twice as long**, and every panel still served. On a *single* night a panel may wait until tomorrow for its turn, so judge a mosaic over the whole run rather than one night.

Which way to go: if a mosaic is stalling and you would rather its panels advance in step, lower it. If a mosaic keeps losing its window to a neighbouring target, raise it. Check the Plan page's Target Summary after a run either way.

Rig overheads

The slew, flip, dither and autofocus overheads, and the autofocus switches, are not set here: they are properties of the rig, so they live on the Imaging System page's **Overheads** sub-tab (Setup shows a one-line pointer with an **Open System > Overheads >** button). They cost real time in every plan, so set them to what your rig actually takes. See [Imaging System](#), including how a slew is charged, and [Analysis](#) for setting them from your NINA logs.

Night Readout

A read-only summary of the night being planned: twilight start and end (labeled by your chosen twilight type, with the UTC offset), **Dark window** in hours, **Moon illumination** and **Night date**.

Tonight's Feasibility

Press **Check** to find out which targets and filters can actually be imaged on each forecast night *before* you spend a run. (It also runs automatically with every schedule.) The summary line says how many targets have problems and how many forecast nights were checked, then lists targets under up to three headings:

- **No feasible time** — the target is never above your altitude floor with a usable filter clear of the moon, on any night.
- **Partly blocked** — some filters, or some nights, are blocked.
- **Available, but never long enough** — the target passes every gate, but never for an unbroken stretch as long as its minimum imaging time, so it loses out every time to targets that do. Common with mosaic panels whose minimum was left at a single-target value.

Objective Parameters section

Two columns: the left tunes Utilization, the right tunes Quality and filter choice.

Utilization Objective

- **Global vs. Per-Target Weight** (default 0.5) — slide toward **Global** to reward filling the night; toward **Per-Target** to reward giving each target a fair share of its own window.
- **Completion Priority** (default 0.5) — from **Off** to **Strong**. Favors finishing nearly-complete targets. 0 turns it off.

Quality Objective & Filter Selection

Filter Preference Rules let you say "when conditions are like *this*, prefer *these* filters". Click + **Add Rule** to add one; click a rule's row to edit it; click X to delete it. Each row summarizes the rule, for example "Seeing ≥ Above Average · L · High (+30%)".

Part	Choices
Condition	Seeing, Transparency, Outside Astronomical Dark, General (always active)
Comparator (Seeing and Transparency only)	At least or At most
Threshold — Seeing	A six-step slider: Cloudy, Poor, Below Average, Average, Above Average, Excellent
Threshold — Transparency	A 0–100% slider (100% = perfectly clear); the sentence under it adds the word label (Poor ... Excellent)
Preferred filters	Any filters from your filter list
Aggressiveness	Low (+10%), Medium (+20%), High (+30%)

A new rule starts as Seeing, **At least**, Average, **Medium**. The sentence under the slider reads the rule back in plain words ("Prefer these filters when seeing is at least as good as Average").

Aggressiveness does two things: it sets how hard the scheduler leans toward the preferred filters when the rule is active, and the strongest rule's percentage sets how much of Quality comes from preference rather than airmass.

- Note:** Switching a rule to a different condition resets its comparator and threshold — a Transparency rule starts at **At most**, the others at **At least**.
- Note:** **Outside Astronomical Dark** only has an effect when Twilight Type is Civil or Nautical. With Astronomical, every planned minute is already astronomically dark, so the rule never fires.
- Note:** A rule that happens to fire all night (or never) under tonight's forecast is fine. It is still doing what you asked.

Dispatch planner rules

Collapsed by default, and read **only** by ► **Dispatch**. Nothing here moves the GA, the other planners or how any plan is scored. The shipped weights are the Target Scheduler defaults.

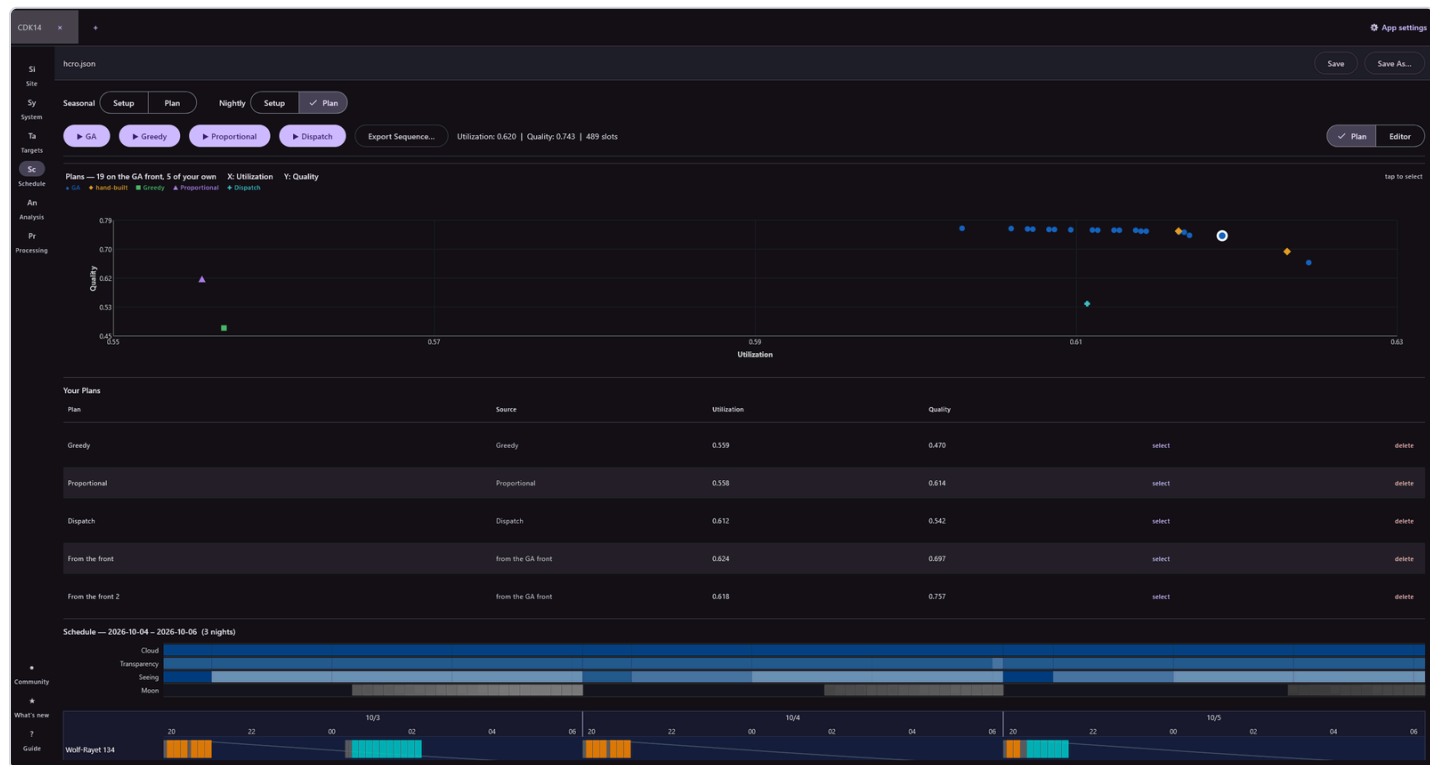
Each rule has a weight slider from 0 to 100. A weight of 0 reads **off** and removes the rule entirely. When a weight differs from the default, its caption shows "(Target Scheduler: *n*)". Scores are summed, not averaged; the section tells you the most any target can total with the current weights (2.17 at the defaults).

Rule	Default	Prefers
Percent complete	50	Targets that are nearly finished.
Priority order	50	Targets higher in the priority order set on the Targets page.
Setting soonest	50	Targets whose window closes earliest tonight.
Stay on target	67	Continuing with the target already imaging, rather than paying another slew.
Avoid meridian flip	off	Targets that won't need a flip during the stint about to start.
Balance mosaic panels	off	Whichever mosaic panel is behind its siblings (opposes Percent complete).
Cloud urgency	off	Targets whose remaining window tonight is about to be lost to cloud.
Seeing urgency	off	Targets whose remaining window is mostly below the seeing floor.
Transparency urgency	off	Targets whose remaining window is mostly below the transparency floor.
Altitude	off	Targets currently high in the sky.
Moon margin	off	Targets whose usable filters are furthest from the moon's required separation.

Heads up: At the default weights, **Stay on target** usually hands one target the whole night, and a mosaic gets finished panel by panel, which is why the Dispatch point often looks lopsided. Set **Stay on target** to 0 to watch the other rules take effect.

Thresholds for the weather rules set what the three urgency rules count as "lost": **Cloud ceiling** (default 30%), **Seeing floor** (default Average) and **Transparency floor** (default 57%, Average). Each urgency rule looks at the target's *remaining* window tonight, so a target about to lose its hours gets imaged first. **Reset to Target Scheduler defaults** puts every weight and threshold back.

Nightly Plan



Making nightly plans

The top row holds the planners and the actions:

- ▶ **GA** — runs the genetic algorithm. The button reads "GA running..." while it works. When it finishes, a balanced point on the front is selected for you.
- ▶ **Greedy**, ▶ **Proportional**, ▶ **Dispatch** — each makes one plan in seconds and adds it to **Your Plans**. Only one runs at a time, and none can start while ▶ GA is running.
- Export Sequence...** — exports the selected plan (see [Exporting to NINA](#)).
- Plan / Editor** — the two chips at the right end of the row. **Editor** opens the [Editor](#) on the selected plan. A point on the Pareto front, or a plan Greedy, Proportional or Dispatch made, is first copied into a new editable plan — "From the front" for a GA point, "Copy of ..." for a generated one — leaving the original where it is; pressing **Editor** again with the same plan selected reopens that copy rather than making another. A plan you built or edited yourself opens as it is. With nothing selected, the Editor opens on the plan it showed last. **Plan** comes back here.

Greedy and Proportional both follow the priority order you set by dragging targets on the [Targets](#) page. With no order stated, every target ranks equally: Greedy then favors whichever target is least complete, and Proportional simply shares the night evenly.

Note: The GA favors fewer, longer visits. Once it starts a target it may stay well past that target's minimum imaging time, so you get fewer slews at the cost of fewer distinct targets per night. There is no setting for this; each target's **Min imaging time** on the Targets page still sets the shortest visit it normally plans.

The buttons stay disabled while targets from a loaded config are still being resolved, and ► **GA** is disabled while another open tab is already running a solve.

Once a plan is selected, the row shows "Utilization: 0.xxx | Quality: 0.xxx | N slots". Errors and notes appear on the same row.

The plan comparison chart

Every plan appears as a point: **Utilization** across, **Quality** up. The title reads, for example, "Plans — 20 of 113 on the Pareto front, 3 of your own". The Pareto front is thinned to about 20 points so it stays readable; your own plans are never thinned out. The chart appears once there are at least two points.

Mark	Color	Meaning
● GA	blue	A point on the GA's front.
◆ hand-built	amber	A plan made in the Editor (blank, duplicated or lifted off the front).
■ Greedy	green	Made by ► Greedy.
▲ Proportional	violet	Made by ► Proportional.
+ Dispatch	teal	Made by ► Dispatch.

Click a point to select it. The selected point gets a white halo, and a named plan shows its name. Up and to the right is better. A point that sits below and to the left of another is simply worse on both counts.

Your Plans table

Every plan other than the Pareto front is listed here with its **Plan** name, **Source** (blank, duplicate, from the Pareto front, or the planner's name), **Utilization** and **Quality**. Click **select** to show it below, or **delete** to remove it (you'll be asked to confirm; it is removed from the saved file too). Plans are saved beside your config and come back the next time you open it.

Stale generated plans are removed

Heads up: A plan made by ► **Greedy**, ► **Proportional** or ► **Dispatch** is removed automatically once the inputs it was built from change — for example your targets, their priority order, the forecast, the number of nights, the twilight type, Min Altitude or the custom horizon, moon profiles, the overheads, and for Dispatch its rule weights and thresholds. A removed plan is gone (there is no undo), and a note tells you: "Removed 1 generated plan — the setup changed since it was built." Press the button again to get a fresh one; it takes seconds.

What is **kept**:

- Any plan you have **edited** in the Editor, even if it started as a generated one.
- Blank, duplicated and "From the front" plans, and copies the **Editor** chip made — those are your work.
- The Pareto front. A solve takes minutes, so it is never removed. If your Min Altitude or custom horizon changes, an amber note says the front describes the previous horizon and asks you to re-run ► **GA**.

A plan restored when you open a config is not removed straight away. Until you change something, it shows a note such as "the setup changed since this was built — regenerate" instead.

Nightly Gantt chart

Titled "Schedule —" and the date (or date range and night count). Above it, a forecast strip shows **Cloud**, **Transparency**, **Seeing** and **Moon** across the night. Below, one row per target — in the Targets page's priority order — with bars colored by filter, overhead in its own color, and a faint altitude trace behind. Nights are placed side by side with the daytime cut out.

Target Summary table

The first line states time on sensor two ways, for example:

Time on sensor: 25.0% of the night's darkness (96.0% of scheduled time)

- **of the night's darkness** — imaging time ÷ all the dark time available. This is the global half of Utilization, and it is the number that shows idle time.
- **of scheduled time** — imaging time ÷ the time the plan actually occupies. This shows how much the overheads cost. A plan that images two hours of an eight-hour night and leaves the rest empty can still read high here.

Before any plan has been scored, only the second figure is shown.

Each target gets a header row reading "Util N% · AM x.xx · Pref N%" — the share of its imageable window used (with ✓ when the plan finishes it), its average airmass, and how often its preferred filters were used. A red **not schedulable** badge means no planner could have imaged it; **min imaging time never met** means it was up but never long enough.

Under each header, one row per filter:

Column	Meaning
Filter	The filter.
Sch/Exp/Req	Frames scheduled / frames expected after the forecast cloud discount / frames still required.
Integration	Scheduled time in this filter.
Avg Airmass, Avg Alt°	Averages over the scheduled frames.
Avg Seeing, Avg Transparency	Forecast conditions over those frames, as word labels.
Pref Match	How often a matching preference rule was satisfied.
Feasibility	Red text: blocked on every night, with the reason. Gray text: blocked on some nights. available, not used : the planner could have used it and didn't. ✓: fine.

Scheduled Blocks table

Every slot in the plan: **Target**, **Filter** (or the overhead type, such as SLEW or DITHER), **Start (local)**, **End (local)**, **Dur(m)**, **Alt°**, **Airmass** and **Frames**.

At the very bottom, **Diagnostics** (collapsed until you open it; the app remembers your choice) shows how the GA's best Utilization and Quality improved generation by generation.

Exporting a plan to NINA

Select a plan, click **Export Sequence...** and choose a folder. NightFront writes:

File	What it is
TargetsForTonight_<date>.json	One NINA sequence per planned night, dated by the local evening the night starts.
TargetsForTonight_<date>.authorship.json	Beside each plan file: who made that night — the GA, a planner (by name), or you by hand.
selection.json	The Utilization/Quality trade-off you picked. Written only when the exported plan is a point on the Pareto front.
session-config.json	A copy of the config the plan came from.

The export uses your targets' current coordinates and rotation, even if you edited them after the run.

Heads up: The NINA plugin reads the authorship file. When conditions trigger a Replan, it leaves a night made by Greedy, Proportional, Dispatch or the Editor **in place** rather than overwriting it with a fresh GA plan. A generated plan you have edited counts as hand-built. Tick **Replan hand-built plans** on the plugin's Replan instruction if you want those nights re-solved anyway. See [Running with NINA](#).

For where these files live and what else sits beside your config, see [Files and data](#).

Editor chip



The Editor is the second of Nightly > Plan's two chips. It is the workshop; the **Plan** chip is still where tonight's plan is chosen. Every plan you build here shows up on the Plan chip's chart and in Your Plans. The **Plan** / **Editor** chips stay at the right end of the top row, so **Plan** takes you straight back.

Managing plans in the Editor

The top row:

- The **plan dropdown** — lists every plan with its source and scores ("0.812 / 0.644"). Pick one to edit it.
- **New blank** — an empty plan.
- **Duplicate** — a copy of the current plan ("Copy of ...").
- **Rename** and **Delete** — act on the current plan.
- **Undo** and **Redo** — step through edits. Each plan keeps its own history, so undoing in one plan never touches another.

Note: A plan is compiled and scored against the session from the last run. After opening a config, the Editor asks you to "Run the GA or a planner (Greedy, Proportional or Dispatch) on the Plan chip first". Your saved plans are still there; they just have nothing to be scored against yet.

While ► GA is running, the Editor shows "Recomputing plans against the new session..." and your plans are rescored against the new run when it finishes.

Reading the Editor timeline

Above the chart: "N block(s) · N frames · utilization / quality", and the priority order a generated plan would use.

- **Rows** are targets, with the night's hours along the top and dates over each night.
- The **green band** along the bottom of a row is where that target is actually imageable (above your altitude floor, with at least one needed filter clear of the moon).
- The **strip above the rows** is the mount's single timeline. Two blocks on different rows can overlap in time and look fine; here they visibly collide.
- A **block outline** is what you declared. The **bars inside it** are what the scheduler actually produced from it: exposures colored by filter, plus overhead. Blank space inside an outline means the block ran out of work or could not use that time.
- A white outline is the selected block; a red outline has a warning.
- A new block with no filter yet reads "assign a filter".

Adding and editing blocks

- **Add a block:** right-click an empty spot in a target's row and choose **Add block here**. The block is as long as that target's minimum imaging time (an hour if it has none), and the filter dialog opens straight away. Tick **Any of this target's filters (what the GA itself does)**, or pick specific filters, then **Apply**.
- **Pick filters:** right-click a block. The menu is the filter list, headed "Filters for ...". It stays open while you tick several. Choose **Any of this target's filters** to let the scheduler use them all. Filters that can't be used are grayed out with the reason: "no remaining demand" (earlier blocks already used it up) or "blocked for part of this block" (moon or altitude).
- **Move:** drag the middle of a block left or right. Blocks move in time only; to put work on a different target, delete the block and add one on that row.
- **Resize:** drag the handle at either end.

- **Snap:** edges land on whole minutes, and stick to another block's edge when you get within 5 minutes of it — the easy way to butt one block against the next.
- **Minimum length:** a block can't be shrunk below its target's minimum imaging time, because the scheduler would refuse to start it.

Several filters in one block are sequenced by the same scored choice the GA uses, restricted to your selection — not a fixed rotation.

The buttons above the chart act on the selected block: **Copy**, **Duplicate** (drops a copy just after it, for you to drag into place), **Paste after** and **Delete block**. The clipboard is shared across plans, so you can copy a block from one plan and paste it into another.

Zoom — **1×**, **2×** or **4×** widens the chart. At 2× and 4× a scrollbar appears under the chart to reach the rest of the night. Zoom is remembered when you switch sub-tabs.

Block details and warnings

Selecting a block shows a card: target, length and filters; then frames produced, minutes of overhead and, when there is some, minutes **unused** — time inside the block that produced nothing. Any warnings are listed in red:

Warning	What to do
starts before the previous block ends	Move one; the mount can only be in one place.
shorter than this target's minimum imaging time	Lengthen the block.
target drops out of view part-way through	Shorten or move it into the green band.
target is not imageable here	Move it into the green band.
nothing left to shoot on this target	Earlier blocks already covered its demand.
ran out of work before the end	The unused time is shown in the card; shorten the block or add filters.
runs past the end of the night	Pull the end back before dawn.
this target is no longer in the session	The target was disabled or renamed; delete the block.
a chosen filter is no longer on this target	Pick the block's filters again.

Note: Editing a plan that came from ► Greedy, ► Proportional or ► Dispatch makes it yours: it is marked "edited by hand since it was generated", it will no longer be removed when inputs change, and it exports as a hand-built plan.

Analysis

NINA writes a log of everything it does. The **Analysis** page reads those logs and tells you what actually happened on your rig: what you collected last night, how long your overheads really take, and, when something went wrong, a report you can send for support. It is a rail page of its own, between Schedule and Processing, with four chips (more forms of analysis will join them):

- **Last night** — the morning report: time on sensor against what the plan counted on, per target and filter.
- **Overheads** — what the logs say your slews, flips, dithers and autofocus runs cost, with an **Apply** button for each setting, and ways to reduce them.
- **Autofocus** — how often autofocus ran and why, whether your three autofocus switches agree with that, and filter offsets.
- **Report** — NightFront errors and anomalies found in the logs, scrubbed and ready to save or copy.

NightFront only ever **reads** NINA's logs, and nothing on this page is sent anywhere.

The night logs

NINA keeps its log open for as long as it runs, so a copy of the live log taken in the morning can stop hours before the night did. Instead, the NINA plugin saves each night's log for you: add its **NightFront Save Night's Log** instruction as the last of your night's end steps (after sky flats, park and warm-up): in the sequence's **End** area, or at the end of the end-of-night block inside a nightly loop, as the plugin's example night does. Each time it runs (the end of each night) it saves the part of NINA's log that belongs to the night, from the noon it began through the morning's flats, into a **nina-logs** folder inside the NightFront data folder. Your config lives in that folder, so NightFront finds the saved nights beside it: there is no folder to choose. The NightFront Plugin Guide describes the instruction itself, under *Instructions and triggers*.

Above the chips, a header shows where the saved nights are read from (the **nina-logs** folder beside your config), with one button:

- **Rescan logs** reads the folder again now.

The plan files are read from the folder your config lives in, where the plugin keeps them.

Until the first night is saved, nothing is read and nothing is written: the header says which instruction to add, and the overhead fields on the Imaging System page's **Overheads** sub-tab behave exactly as they always have: you type the values. A config that has never been saved has no folder beside it yet, so save it first. You can still analyze log files you pick by hand (see **Choose files...** below).

What the saved nights keep. Once a night is saved, NightFront reads the folder when the config opens, when the Analysis page opens and when you click **Rescan logs**, and adds only what it has not seen before, so reading the same nights twice changes nothing. A night is stored only after noon the next day, so a night still being imaged is never counted half-finished. The record sits in a file beside your config, and NightFront also copies each plan file it finds before the NINA plugin tidies the originals away a few days later, so a clear night can always be compared

with the plan it ran. Because the record is yours, it survives NINA deleting old logs. The files are listed under Sidecar files beside a config in [Files and data](#).

If you used an earlier version that read a NINA logs folder you chose, the measurements it kept stay, and a night already kept is not counted again when its saved log arrives.

Reading the folder is cheap and automatic. **Analyzing** it, which is what the chips below show, is never automatic: nothing is analyzed until you press a button.

Choosing which nights

Under the header, one row decides which saved nights the page uses, and **every chip uses the same nights**: Last night, Overheads, Autofocus and Report.

- **Most recent [N] nights** takes the newest N saved nights. Type the number and press Enter (or click away). It starts at 7, enough nights for most of the Overheads recommendations. The Last night chip still shows only the newest of them unless you ask for all (see **Last night** below).
- **Picked nights** uses exactly the nights you tick. **Pick nights...** opens a checklist of every night with a saved log, newest first, with the number of saved logs for each night and, once NightFront has kept that night, how many LIGHT frames it collected. **Select all** and **Clear** help with a long list; **Use these** switches to the ticked nights, and **Cancel** changes nothing.

Choosing a count replaces a picked list, and **Use these** replaces the count. The choice is kept while you move around the app, for each open config, and starts again at the most recent 7 nights when the app restarts or the tab opens another config.

The unit is a **night**: every saved log whose name carries that night's date, so a second copy of the same log (say, one a sync tool left beside it) is read with it and no frame is counted twice. Only nights with a saved log in the **nina-logs** folder are offered. Measurements an earlier version kept from a NINA logs folder you chose are not in the list, so they no longer count toward any chip.

Then press **Analyze** to read the chosen nights. Each night's log holds the whole night, flats included, so a single night takes seconds; months of nights take a while. With no night saved yet, it says which plugin instruction to add. **Choose files...** shows pickers for individual log files and plan files instead, so you can analyze any log you pick by hand, NINA's own included, with or without saved nights. Each picker has two buttons: **Choose files...** picks one or more files and **Choose folder...** picks a whole folder (every log in it, or every plan file in it and its replan history). Both add to the list, so you can combine files from several folders; **Clear** empties it. The dialogs are your system's own. The read's progress, and **Cancel**, show under these buttons whatever chip is open.

Last night

The morning report. The page shows "Last analyzed" with the time, and what was read. Two chips at the top choose what follows: **Last night** (the default) shows one block, for the newest of the nights you chose; if you picked nights that leave out the latest one, it is the newest night you picked, and its heading names the date. **All nights** shows

one block per chosen night, the newest expanded and older nights collapsed. The other chips always use every chosen night, whichever is selected here. Each block holds:

- **Time on sensor, predicted and actual**, and the night's darkness. The predicted figure comes from the plan files: the ones in your config's folder plus the copies the store kept. A night with no surviving plan reads *Actual only: no plan kept for this night*. The NINA plugin removes saved plans after about four days, which is why NightFront keeps copies. When the plugin replanned during the night, every plan is added up, each for the hours it was in charge: the evening's plan until the first replan, each replan until the next, and the last one to the end of the night. The line above the table names the plans compared.
- **What was collected, per target and filter**, against what was planned. A frame's filter is the last filter change that started before it; a frame after a failed or interrupted move is left unattributed rather than guessed.
- **A flats line** counting FLAT frames per filter, from the log. It states only what the log shows.
- **A NightFront problems line** counting the errors and anomalies found, with a link to the Report chip. Under **Last night** it counts the newest night's problems and, when the Report holds more from the other chosen nights, says how many are in the whole report; under **All nights** it counts them all.
- A night that was still being imaged when you analyzed it is labeled **In progress**; analyze again in the morning for the whole night.

Before you press **Analyze**, the chosen nights show the totals the store keeps (time on sensor, frames, darkness) and no per-target table, because the store keeps nothing per target.

Overheads

What NINA's logs say about this rig's overheads, and what to enter on the Imaging System page's **Overheads** sub-tab instead of guessing. It works out what each overhead really took, from the timestamps NINA wrote. The page leads with two headed sections; everything else is tucked away under **Supporting evidence**.

Setting recommendations

Once a night has been saved, the main table is built from the measurements the store kept for the nights you chose above, counting only the stretch since the most recent change that could move each setting (see **Eras** below). One night rarely holds enough samples for every setting, so with too few nights chosen some rows are greyed, and a note under the table says to raise **Most recent** or pick more nights. One row each for Max Slew/Center/Rotate, Meridian Flip, Dither and Autofocus Run: your current value, the recommended one, how many samples it rests on, and their median, 75th percentile, maximum and number of failures. A row with too few samples is greyed out, says why, and cannot be applied. The slew row stays informational until enough centrings of the right kind have been logged. **Edit on System > Overheads >** jumps to the sub-tab where those fields live.

Apply, one row at a time. Nothing changes until you click **Apply** on a row, and then only the open config changes, in memory. **Save** highlights as for any other edit; nothing reaches disk until you save. A recommendation equal to your current value reads *Applied*, with no button. The Apply here and the one beside each field on the Overheads sub-tab are the same action.

A discard in force is footnoted under the table, with the cutoff and what it hid, so a table resting on fewer nights than the logs hold says why.

Ways to reduce overhead

Concrete fixes, each with the minutes per night it would have saved and what to change in NINA. NightFront changes nothing in NINA; these are suggestions for you to act on. The section is always there: before an analysis it reads *press Analyze at the top of the page*. The suggestion that NINA use filter offsets is a one-line pointer to the **Autofocus** chip, where it is shown in full.

Supporting evidence

One group, closed until you open it (and kept as you leave it when you switch pages):

- **Overhead settings from the analyzed files**, with the percentiles, labeled as computed from those files alone and never mixed with the kept measurements above.
- **Slew detail**, by the size of the move.
- **Where the time between frames goes**. An inventory of every kind of overhead, with the time that no setting reserves marked as such.
- **Centring**. How many slews, rotations and plate solves a typical centring took.
- **Plan against what happened, recent nights, and before and after each change**, with the era tools (see below).

The recommendations describe what your rig did, so they are only as good as the nights behind them. Fewer samples mean a less certain number, which the sample count shows.

Eras, and plan against what happened

When something that changes how long things take is changed, NightFront notes the date and starts a new **era** for only the measurements that change can affect. An era starts when you Apply a suggestion, when you edit an overhead value by hand, and when a NINA setting that the logs state changes (the rotator, the centering tolerance or the plate-solve exposure). Nights before the change no longer count toward the suggestions it affects, so an old habit does not drown out a fix.

Within **Supporting evidence**:

- **Plan against what happened**. A line comparing the time on sensor the plan counted on with what NINA collected, over the last 10 imaging nights. A night with no kept plan shows only what was collected. A recent-nights column sits beside the figures since the last change, so a fix is not diluted by older nights.
- **Before and after each change**. For the newest changes, both sides of the change side by side. A side with too few samples or nights is left out rather than guessed.
- **Start a new era on...** Records a change the logs cannot state themselves, such as a new focuser or the rotator's mechanical range, with the date. Choose what it applies to: everything, slews and flips only, or the nightly overheads only. For the rotator's mechanical range start there is a one-click fill.
- **Discard samples before...** Pick a date, and everything older stops counting. This holds even when the same logs are read again, so a folder that still contains old nights does not bring them back.

On the Imaging System page's **Overheads** sub-tab, each overhead field shows the same suggestion as a one-line hint, such as *logs suggest 0.4 min (from 25 dithers, since 1 Oct 2026) — now 0.5 min*, with its own **Apply**, and a **See**

Analysis > Overheads > button that jumps here. A setting with too few samples shows no hint. The hints use the nights chosen on this page, and the sub-tab says which.

Autofocus

How autofocus really ran on your rig, and whether the three autofocus switches agree with it. Like the Overheads chip, it leads with two headed sections and keeps the detail in one closed group.

- **Autofocus switches.** One row each for *Periodic*, *On every filter change* and *After every slew*: whether it is on in your config, what the logs show (how often autofocus ran for that reason), what NightFront recommends, and an **Apply** to change it. A switch the logs agree with reads *Consistent*, with nothing to apply. The suggested periodic interval is the average gap between the autofocus runs the scheduler cannot predict; applying it sets the interval and the switch together, and turning periodic off leaves the interval alone. There is no switch Apply at all without an imaging night in the analysis, because with none every cause reads zero. **Edit on System > Overheads** > jumps to where the switches live. How long one run takes is the **Autofocus Run** row on the Overheads chip.
- **Filter offsets.** When NINA refocuses on every filter change, the suggestion is NINA's filter offsets instead (a stored focus offset for each filter, *Use filter offsets* on, no filter flagged as the autofocus filter, and the *Autofocus after filter change* trigger removed at the same time you turn offsets on, never before), with the autofocus runs and minutes that would go away. Where the logs show where each autofocus settled, it also estimates each filter's offset from the reference filter; that estimate comes from pairs of autofocus runs a few minutes apart, not bracketed measurements, so treat it as a rough starting point. Beneath it, a standing panel shows whatever the NightFront NINA plugin's *NightFront Filter Offsets* instruction measured, whenever `<config-name>.filter-offsets.json` sits beside your config, with or without an analysis.
- **Supporting evidence** (closed): the autofocus runs by cause, the periodic interval per night, and the startup runs.

With no analysis and no filter-offsets file the page says so in one line and tells you what to do.

Report

A report for NightFront support: any NightFront errors and anomalies found in the logs, gathered into one text that is scrubbed of your user name, machine name, equipment serial numbers, site coordinates, keys and email addresses. It comes from the current analysis, so with none it reads *Press Analyze at the top of the page first*. It is shown to you in full first; use **Save report...** or **Copy**. NightFront never sends it anywhere, so you decide whether and how to pass it on.

Below it, a note on crashes in NightFront itself. A crash or a window that vanished is not in NINA's logs: NightFront writes its own errors to `nightfront-err.log`, and that file is the first thing to send when you report one. See [Troubleshooting](#).

Processing

The **Processing** tab is where a night's lights turn into accepted integration time and, eventually, into stacked masters. It works on your own folder of projects, drives PixInsight or Siril unattended for calibration and stacking, and writes the result back into the one number the planner cares about: how much of each target's time you have actually accepted.

Everything here is app-side — the NINA plugin knows nothing about this tab, and none of these settings travel with the config. Calibration, master building and stacking normally run on the machine you're using, but any of the three can be sent instead to another NightFront on your LAN set up as a processing server; see [Remote processing](#).

The closed loop

Processing exists to close a loop that runs for weeks per target:

1. **Plan and schedule** the target — in the app or unattended through the NINA plugin.
2. **Collect** lights and calibration frames.
3. **Import** a night's lights from your download folder into each project's `Inbox\`.
4. **Blink** them (optional) and throw out the obviously ruined ones.
5. **Organize** the project into sessions.
6. **Calibrate** the lights against your master library.
7. **Review** the calibrated lights against their measured statistics and reject outliers.
8. **Accept** what remains — this is what updates the target's accepted time.
9. Repeat 2–8 until there is enough to stack.
10. **Stack**, with frame selection before registration.
11. Read the **rejected-frame ledger**: it says which frames did not reach a master, where and why. Remove the bad ones and accept the corrected count.
12. Go back to scheduling until the target has enough, then **Complete** the project.

Why accepted time matters. The accepted count on each filter is what tells the planner how much a target still needs. It is not a readout; it is what you and the scheduler steer by. A cloud-ruined sub that stays counted tells the planner that filter is closer to done than it is, and the target quietly stops getting time while its data isn't actually on disk. That is why rejection and acceptance are separate, reviewed steps, and why Processing changes the count only when you press **Apply** on an accept proposal.

The Projects tree on disk

Processing reads a folder you choose — the **Projects root** (set in **App settings > Paths**) — laid out one level per imaging system and one level per project:

```

<Projects root>\
└─ <rig folder>\
    │ nightfront-processing.json    one per imaging system
    │ Calibration Library\         this rig's links, annotations and settings
    │ │ Archive\                  the rig's master library (any sub-folders)
    │ │ Completed\               subs of master flats NightFront built
    │ │ archive\                 finished projects, no longer scanned
    │ │ <project folder>\        never scanned
    │ │ │ Inbox\                 one per target or mosaic
    │ │ │ Originals\             imported lights nobody has organized yet
    │ │ │ │ rejected\            your lights (Lights_Session_N\ after Organize)
    │ │ │ │ Calibration\         lights you rejected – never scanned
    │ │ │ │ Calibrated\<filter>\ project-local masters and the copies Organize makes
    │ │ │ │ Stacked\             calibrated lights (_c files)
    │ │ │ │                     stacked masters

```

A mosaic mirrors the same shape all the way down, with a folder per panel in each tree:

```

└─ <mosaic>\
    │ Originals\<panel>\Lights_Session_N\
    │ Calibrated\<panel>\<filter>\
    └─ Stacked\<panel>\

```

- Lights are only ever read from `Originals\`. Frames in `Inbox\` are never counted, calibrated or stacked until Organize moves them.
- The ignored folder name (`rejected` by default) can be changed per rig on the Settings sub-tab's **Rig** chip. A rig laid out the old way can also change the name of its completed-projects folder; a rig using the managed layout below always uses `Completed\`.
- Each project folder carries a small hidden `.nightfront-project.json` and each rig folder a hidden `.nightfront-rig-id`. They are how NightFront recognizes a folder after it has been renamed, even in File Explorer — see [Files and data](#). Leave them in place.
- A project folder is matched to a target in this config by a link you made with **Link...** first, then by the lights' `OBJECT` header, then by folder name. A link follows the target itself, so it survives renaming the target. Mosaic panels map to the mosaic.
- Your processing work — which config a rig folder serves, hand-picked masters, annotations, review records — is stored in `nightfront-processing.json` inside the rig folder, so it travels with the data. Caches are kept on this machine only.

The managed layout

A rig folder can keep its projects in four fixed folders instead of loose at its root:

```

<rig folder>\
└─ Active\           projects whose target is Active
└─ Planned\          projects whose target is Planned
└─ Unlinked\         folders no target owns any more
└─ Completed\        finished projects, no longer scanned

```

The project folders themselves look exactly as above. `Active`, `Planned`, `Unlinked` and `Completed` are reserved names — a project can't be created or renamed to one of them — and a project name can be used only once across `Active\` and `Planned\` (the same name in `Completed\` or `Unlinked\` is fine, for a re-shoot). A rig laid out

the old way works exactly as it always did until you choose **Adopt managed layout...** on the Settings sub-tab's **Rig** chip (see [Processing Settings sub-tab](#)); a new rig folder made with **New rig folder...** is managed from the start.

In a managed rig, project folders follow your targets — but only from the config **as saved**:

- **When it happens.** After you open a config, again after each **Save**, and on a **Rescan** that finds a folder out of step with the saved config — renamed or moved by hand in Explorer, say, or missing — which is put right at once, with no Save needed. An edit you have not saved moves nothing, and a rename or a move between Active and Planned shows "rename pending — save to apply" (or "move pending") on the target's card until you do.
- **A new target** gets a folder, in `Active\` or `Planned\` to match it.
- **Moving a target** between Active and Planned moves its folder between the two.
- **Renaming a target** renames its folder — but only a folder NightFront created, or one you accepted renaming when adopting the layout. A folder you named yourself keeps its name.
- **Deleting a target** moves its folder to `Unlinked\`; nothing is deleted. Link it to another target from the **Unlinked** chip if you want it back in use.
- **Completed is never automatic.** NightFront never moves a folder into or out of `Completed\` except through **Complete**, **Restore** and the Targets page's **Plan again**.
- **Folders NightFront leaves alone:** a folder in a rig folder other than the primary, or loose at a managed rig's root. The target's project line says so.
- **An empty copy at the root is not a project.** A folder loose at a managed rig's root that has the same name as a project in `Active\`, `Planned\` or `Unlinked\` and holds no frames at all — typically the old copy of a folder that was moved, put back by a file-sync client — is listed on the Projects sub-tab, above the project list, as a leftover instead of as a second project, and never claims the target. If it holds nothing but empty folders, **Remove...** deletes it after you confirm; if it holds any other file, the note lists them and you delete it yourself. If a sync client mirrors the Projects folder, delete its local copy first, or it will upload the folder again.

Every move is listed in a **Folder changes** note under the file bar, which stays until you dismiss it. A move is never forced: when a file in the folder is held open, frames are waiting in `Inbox\`, or a job sent to a processing server is still using it, the move is skipped, the note says why, and it is tried again the next time you open or save. A skipped move also shows as a warning on its target's card. When you press a button on a project's card and the action is refused, the reason appears under that card's buttons (it is repeated in the status area at the top of the page, which clears the previous action's summary when you start the next one). When NightFront can't tell which folder belongs to a target — several folders carry its name, or one was linked to a target that no longer exists — it asks on the target's card instead of guessing.

Linking a rig folder

A rig folder serves this config automatically when its name contains the config's imaging-system name as a whole word (set on the [Imaging System](#) page) and no other config has claimed it. Otherwise open **Other rig folders** on the Settings sub-tab's **Rig** chip and tick it. The **Rig folder** picker there chooses which of the folders serving this config is its primary one — the one holding the Calibration Library and the master builds, and the one new projects land in — and **Rename...** and **New rig folder...** beside it rename or create one. Renaming a rig folder, in NightFront or in File Explorer, does not make NightFront forget what it has calibrated there.

Note: One rig folder belongs to one imaging system. Calibrated frames and run records are stored per folder, so two systems sharing one could keep re-calibrating each other's lights. Ticking a folder another config already claims asks you to **Take ownership** first.

Processing sub-tabs

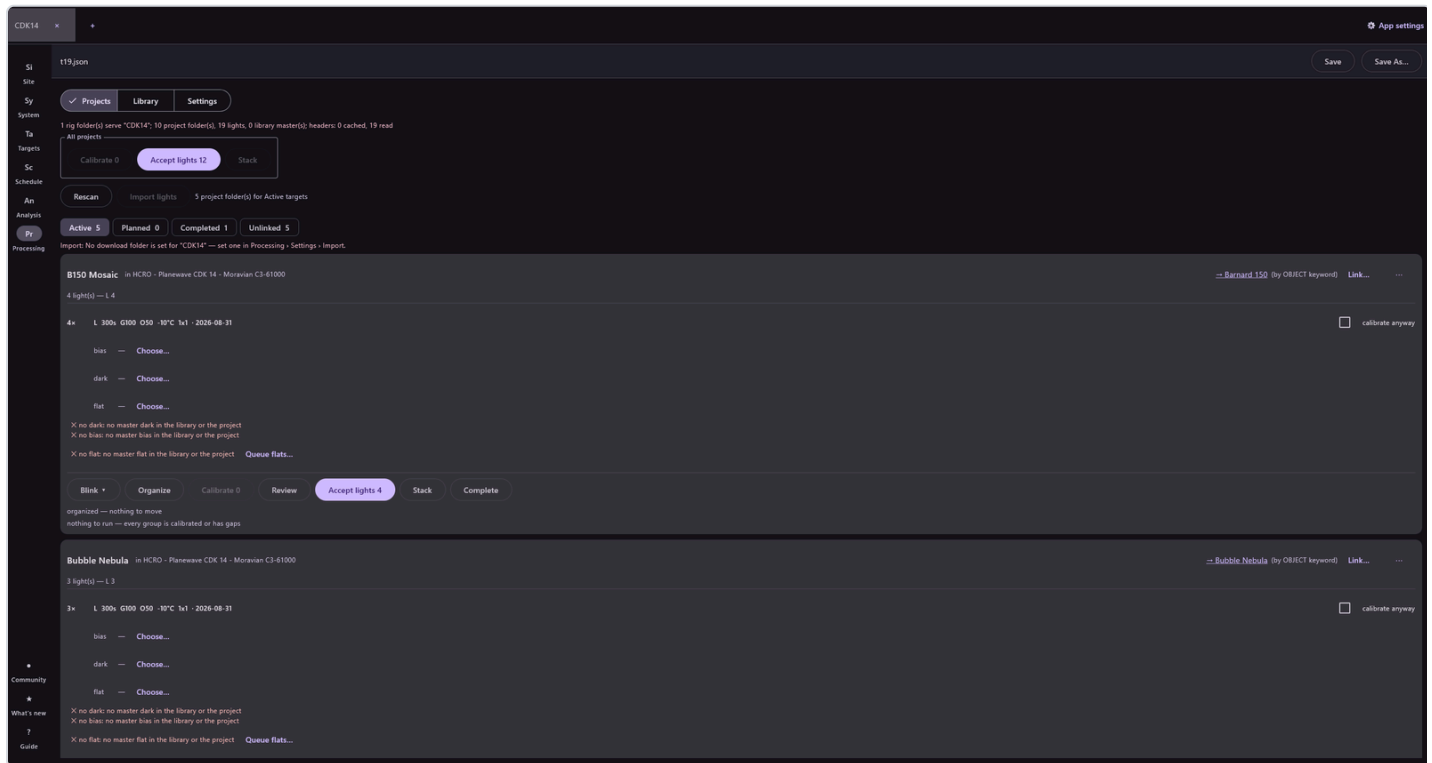
Sub-tab	What it's for
Projects	This config's project folders, each with its calibration matches, and the workflow buttons
Library	The rig's master calibration frames, how each was identified, and building new masters
Settings	Chips Rig , Import , Calibration and Stacking : the rig folder, this imaging system's import and calibration settings, and its stacking recipe (the engine and program paths are in App settings)

Above all three sub-tabs sits a status area, so something started on one page still reports while you look at another: a spinner with **Cancel** while something runs, the per-stage report of each run (failed stages in red), the rejected-frame ledger after a stack, and **Show engine output** — which appears only when a stage actually failed. A local run and a job sent to a processing server (see [Remote processing](#)) each get their own panel, so running one beside the other never mixes their reports or their rejected-frame ledgers. When a run ends, its panel collapses to one line — what ran and how it came out — with **Show details** to open it again and **Dismiss** to remove it; a run that failed stays open so you see its error. With two or more ended runs on screen, **Dismiss all** clears them in one click. An ended panel stays until you dismiss it or start the next run.

Scanning is not automatic except once: the first scan starts as soon as a config is opened with a Projects root set, without waiting for you to visit this tab. After that, press **Rescan** when the tree has changed. A scan over a network share can take a minute or two.

Only one run (calibration, a masters build, a review measurement, a stack, a completion) can happen at a time across the whole app, including other open config tabs. Buttons that would start a second one are disabled and the status line says what is busy.

Projects sub-tab



A bordered **Active projects** box sits above the header row (see [Running an action over every active project](#)); the header row itself is:

- **Rescan** — re-reads the tree.
- **Import lights N** — copies a night from your download folder (see [Importing lights](#)). Filled, with a count, when frames are waiting; the count refreshes every 30 seconds while the page is open. When importing isn't possible, the reason is printed under the row.
- **Status chips** — **Active**, **Planned**, **Completed** and **Unlinked**, each with a count of its projects, whether selected or not. Select any combination to list those projects together; **Active** alone is what you see to begin with, and your choice is kept while you move between pages. A project sits under the chip of its *target's* state (**Planned** for a Planned target's folder), which can differ from the folder it is in until the config is saved and the folders have followed. **Unlinked** lists folders no target of this config owns, where you can still **Link...** one; **Completed** lists finished projects, where **Restore** is; each row shows its target as a link that jumps to it in the Targets page's Completed strip (a folder with no stored target link shows none).

Running an action over every active project

The **Active projects** box holds three buttons, in the same order as the per-project card below:

- **Calibrate N** — calibrates every active project that has uncalibrated lights, one at a time.
- **Accept lights N** — one accept proposal pooled across every active project (same rule as the per-project **Accept lights N**, just totaled).
- **Stack** — stacks every active project that has anything to stack.

All three read the same projects: those the **Active** chip lists, whatever chips are selected. A project whose target is Planned, Completed or unlinked is never included, and **Calibrate** and **Stack** skip a project with nothing to do. While either is running, a status line under the box shows which project it is on ("*Stack all — 3 of 7: Stacking M 31...*") with its own **Cancel**, and a line naming that project is added below it as it finishes, fails, or is skipped. A project that fails does not stop the batch; the next one still runs, and its own line says what happened. Once every project has had its turn, the status line itself becomes a one-line finish ("*Stack all — done: 3 projects*", or "*Stack all — cancelled after 1 of 4*" if you clicked Cancel while a project was running) — it does not stay on the last project it was working on. If you cancel while the very first project is still waiting for a busy processing peer, nothing was sent anywhere yet, so the line instead reads "*Stack all — cancelled before any project was sent*".

Stack asks for confirmation first, because it runs **unattended** — every question a stack could normally stop and ask you uses a default instead, so you can start it and walk away:

- **Frame selection runs automatically**, on each project's stored limits, even for a project whose setting is Manual review. This is for this run only — an ordinary, single-project **Stack** on the same project right afterward still opens the review window. (Siril has no frame selection at all, so this bullet does not appear when Siril is your calibration engine.)
- **A second OBJECT name found among a project's frames stacks as its own group**. Nothing is recorded about it, so the next single-project stack of that project still asks you what to do with it.
- **A thin master is left alone rather than replaced**, and the run tells you it did — naming, in a batch, the project's own **Stack** button (which still asks) rather than a tick, since a batch has no tick to offer.

The confirmation lists every project about to run, where each one's masters will land, and these three defaults in plain words before you press **Start**.

Uncalibrated lights are calibrated first, on every Stack. This applies to the per-project **Stack** button too, not only Stack all: press **Stack** and it calibrates anything not yet calibrated, then stacks — one action, skipping the separate Review step in favor of the frame selection built into the stack itself. The per-project **Calibrate** and **Review** buttons are unchanged; this only changes what **Stack** does on its own.

Reading a project card

- **Header** — the folder name, the rig it lives in, and the target it maps to with how it was matched: *by OBJECT keyword, by folder name or linked*. The target's name (→ **M 42**) is a link: it switches to the [Targets](#) page, opens the column holding that target, scrolls to its card and expands it (a mosaic and its one project link to each other the same way). **Link...** picks a target by hand — its menu lists every target (Active, Planned and Completed, under a heading each; a mosaic once); **Unlink** removes a hand link. A hand link wins over what the **OBJECT** header and the folder name say, and keeps working after the target is renamed. (A link made by an earlier NightFront version works as before: it takes effect only when neither of those already matches a target, and a rename breaks it — re-link it to update it.) A folder that matches nothing says *not a target in this config* and lists what the lights' headers say.
- **Counts** — lights per filter, project-local masters, copies in `Calibration\`, frames waiting in `Inbox\`, online-only files skipped, and unreadable files.
- **Sessions** — collapsed rows, one per session (see [Organizing sessions](#)), each showing its flat set, its nights, its light count and what an Organize would still move. Expand one to see its flats, its lights by folder, and its light groups.

- **Light groups** — one row per group of lights that share a calibration key, for example `42x Ha 300s G100 010 -10°C 1x1 · 2026-08-01 ... 2026-08-09`. Under it, one line each for **bias**, **dark** and **flat**: a ✓ and the full file name of the matched master, with where it lives (*library*, *project-local*, *external*, or *chosen by hand*).

Calibration matches and gaps

NightFront matches masters to lights by the rules you'd apply by hand: bias and dark masters by gain, offset, readout mode, binning and temperature set-point exactly (darks by exposure too); flats by filter, gain, offset, readout mode, binning and rotator angle ($\pm 1^\circ$), then the flat nearest in date *for each light*, so one group can split into rows by flat. A matched dark carries the bias.

When no master fits, the row shows a red ✕ **no bias / dark / flat** line with the reason. You can:

- Tick **calibrate anyway** to calibrate that group with the masters it does have.
- Use **Choose...** on any master line to pin a specific master — the list puts masters that fit the group's key first, library first — or **Browse for a file...** A hand pick is labeled *chosen by hand*; **Clear override (back to the rules)** removes it.
- On a flat gap, **Queue flats...** adds a calibration requirement that the NINA plugin's flats instructions will shoot (see [Running with NINA](#)). It needs a plan to have been exported from this machine once, and it shows *flats queued* or *flats shot* afterwards.

The step buttons

Each card has a row of workflow buttons in loop order: **Blink N**, **Organize**, **Calibrate N**, **Review**, **Accept lights N**, **Stack** and **Complete**.

Exactly one of them is filled: the earliest step that has work you can act on. Later steps with work stay usable; only the emphasis is singular. A step you *can't* act on in this setup (no engine path, Review under Siril, no linked target) is skipped rather than blocking the rest. Under the row, lines say what is outstanding — for example *12 light(s) not yet reviewed · 40 not yet in a master* — and where calibration will write.

Stack and Calibrate follow what is actually in the project's folders, not just what has run before. Delete a master (or the whole `stacked\` folder) and Rescan, and **Stack** lights again for the frames it used — you don't need to re-run anything else first. Reject a frame that already reached a master, and **Stack** lights up for that master; if any of its other, still-good frames no longer have a calibrated copy on disk — you've since deleted `calibrated\` to free space, say — **Calibrate** lights up for those too, so the re-stack has the whole unit to work with. A line under the row says why, for example *"Stack: the Ha master was made with drizzle off"* or *"Calibrate: re-stacking Ha needs its 108 older frames calibrated again."*

Importing lights

Set a **Download folder** for this imaging system on the Settings sub-tab's **Import** chip first — the folder your imaging computer drops a night into (a share, a cloud-synced folder, a USB stick or a local folder). Exactly one rig folder must serve the config.

1. **Import lights** opens a preflight: how many files and how many bytes will be copied, by extension, and anything skipped. Files whose names positively read as flats, darks or biases are skipped by name and listed; everything else is copied and checked by its header.
2. **Copy N files** copies each file into the rig's `.intake\` staging folder, reading every copy back to prove it is byte-identical before going further.
3. A routing window groups the lights by their `OBJECT` header and sends each group to its project's `Inbox\`. A mosaic is **one** group, not one per panel: its panels carry different `OBJECT` names but they are one target, so they are listed together — headed by the mosaic's name and the panels it covers — and go into a single project folder. Groups that match no project can be given a new folder with **Create project** and a **Project name**, or sent to **Assign to existing...** Frames that were already imported are listed with where the earlier copy is (`Inbox\`, `Originals\`, `rejected\` or a completed project).
4. **Import N lights** places them. An unticked group's staged copies are discarded and its originals in the download folder are left alone, so it is offered again next time.

Anything that isn't a LIGHT is refused by name after the copy. A cancelled import can be resumed: leftovers in `.intake\` are picked up by the next run, and the button offers to **Place N staged files** even when the download folder is empty.

Heads up: Delete each file from the download folder after importing it is off by default. If you turn it on and the download folder is synced or a cloud drive, the delete removes the file everywhere it syncs, not just from this machine. A source is only ever deleted after its copy was proven identical and placed in a project's `Inbox\`.

Blinking a night

Blink N opens a viewer window over the frames in the project's `Inbox\`. It works without PixInsight or Siril installed. Blink is optional and gates nothing.

A night's lights open **grouped by filter** — one filter's frames at a time, in the order they were taken, filters in filter-wheel order — so consecutive frames are directly comparable instead of alternating filters the way NINA shot them. Press **F**, or the **By filter · By time** segment in the status line, to switch to plain time order for the whole night (useful for spotting something that crossed every filter, such as a cloud); the segment only appears when the two orders actually differ. Your choice is remembered on this machine and used the next time you open Blink. Switching orders keeps every ruling you have made.

Every frame opens accepted, so the job is naming the exceptions. A frame is either accepted or rejected — there is no third choice; if in doubt, keep it and let frame selection filter it later. The toolbar has **Stretch (Auto / Linear / Locked)**, **Strength**, **Zoom** (Fit · 25 % · 50 % · 100 % · 200 %), **✓ Accept**, **✗ Reject**, **★ Reference** and **Commit...** Rejected frames show dimmed in the filmstrip. The status line shows the frame's statistics, a clipped-cell count, the tally, and a small curve labeled *sky* — each frame's background level, scaled to that night in time order, or with a gap between filters and each filter scaled to itself when grouped. Grouped by filter, the status line also carries **← Group** and **Group →** (the same as PgUp and PgDn) with a label between them naming the filter you are viewing and your place within it — for example **Ha · 12 of 40**. A night of one filter shows the label on its own; in time order the label and the group buttons go away, leaving their space empty so the **By filter · By time** segment stays exactly where it was. If you set the reference frame on one filter and then move to another, the status line flags that the reference is a different filter. On a narrow window the MAD, min, max and clipped-cell figures are the first to

make room, and the tally shortens to **32✓ 0X** (accepted, rejected — hover over it for the full wording) rather than disappearing.

Key	Action
← / → , Home / End	Previous / next, first / last frame
A · X · Space	Accept · reject · switch between accepted and rejected
F	Switch between grouped-by-filter and time order (a night of one filter has no F to press)
PgUp · PgDn	Jump to the start of the previous · next filter group (grouped order only)
R	Set or clear the reference frame
Tab (hold)	Flip to the reference while held
S	Cycle the stretch mode · L locks the stretch to the reference
[·]	Loosen · tighten the stretch
+ · - · 0	Zoom in · zoom out · fit
Enter · Esc	Commit · close

Verdicts are saved as you go, so closing the window and coming back keeps them. **Commit...** shows where the frames go: accepted ones to `originals\` and rejected ones to the rig's ignored folder (`rejected\` by default), so every frame you saw leaves `Inbox\`. Nothing is deleted, and **Cancel** reopens the window with your verdicts intact.

Heads up: A frame that reached `Inbox\` after you opened the window was never shown, so the commit leaves it there and says so. It stays in `Inbox\` only until the next **Organize**, which moves everything in `Inbox\` into `Originals\` whether or not it was blinked — blink again first if you want to look at it. The Organize confirmation says how many unblinked frames it will count.

Blink needs the rig to name an ignored folder (`rejected` by default); without one it is disabled with the reason under the row.

Blinking lights already in Originals or Calibrated

The ▾ beside **Blink** opens a menu — **Blink Inbox** · **Blink originals...** · **Blink calibrated...** — for looking back over lights you already organized, without waiting for a fresh night's Inbox. With an empty Inbox the whole button reads **Blink ▾** instead of being disabled, and its own click opens this menu directly.

Blink originals... and **Blink calibrated...** each open a dialog first, with two independent checklists — **Filter** and **Session** — both ticked in full; untick either to narrow what **Blink N** opens (a light needs both its filter and its session ticked to be included). **Blink calibrated...** shows each light's calibrated copy in place of the original, so you judge the pixels the stacker actually uses; a light is left out — and the dialog states how many and why — when it has no calibrated copy yet, when its copy is ambiguous (calibrated by both engines) or not on this machine, or when it was calibrated from a color sensor and can't be previewed once debayered.

The window that opens is the one described in [Blinking a night](#) above, with the same keys, but it behaves differently at the two ends:

- **Accepting does nothing** — an original or calibrated light is already where it belongs. **Rejecting** parks the original in the rig's ignored folder and discards its calibrated copy, in either scope, and the light leaves the project; any master that was built from it goes stale, and **Stack** lights up for that unit as soon as the commit's rescan finishes.

The confirmation before committing states what will move, names any current masters a rejection makes stale, and lists any rejects it could not carry out, with the reason. **Under Siril, committing a Calibrated blink also counts as reviewing every light it showed** — accepted or rejected alike — the same as running Review, so their Review badge clears without a separate pass. Committing an Originals blink does not: it shows the acquisition, not the calibrated pixels Review measures.

Organizing sessions

A **session** is a run of consecutive flat nights in which no filter repeats — the fewest sessions possible, but no fewer. **Organize** makes the project folder stand on its own:

- moves light files into `Originals\Lights_Session_N\` (a mosaic: `Originals\<panel>\Lights_Session_N\`), including anything waiting in `Inbox\`;
 - A project is a mosaic when the **target** it maps to is one — not when its frames happen to carry more than one `OBJECT` name. So a mosaic gets a panel folder from its very first panel, before the others are shot, and a single target whose frames carry two spellings of its name does not get one at all. The same rule decides `Calibrated\<panel>\` and `Stacked\<panel>\`, so the three trees always agree.
- copies the matched masters into `Calibration\Flats_Session_N\`, `MasterDark\` and `MasterBias\`;
- gathers rejected lights into one `rejected\` folder;
- removes flat copies no light uses any more, and retires an older copy of a master when a rebuilt one replaces it.

Existing sessions keep their numbers; if two of them now belong together they merge into the lower number. A light that matched no flat is left where it is. A confirmation lists every session and every count before anything moves.

Heads up: Organize **moves** your captured lights. Calibration history follows them (matched by each frame's capture time), so already-calibrated frames are not redone.

Calibrating lights

Calibrate N — N is the number of lights — runs the selected engine headless over every group that is fully matched (or ticked **calibrate anyway**), batched by master set. Output lands in `Calibrated\<filter>\` with a `_c` suffix. A light that was already calibrated successfully, and whose calibrated file is still in the tree, is skipped, so pressing Calibrate again only does the new work. Delete a calibrated file to have its light redone.

The engine path must be set in **App settings > Paths**, or the button is disabled and the card says *Set the PixInsight executable in App settings > Paths* (or the Siril equivalent).

Cosmetic correction — hot- and cold-pixel correction, and an optional defect map, configured once per imaging system on the Settings sub-tab's **Calibration** chip, in its **Image Calibration** section (see [Cosmetic correction](#) below). With the section left at its defaults, calibration behaves exactly as before.

Reviewing calibrated lights

Review measures a project's calibrated lights (FWHM, eccentricity, stars, PSF signal weight, SNR and median) so you can reject outliers *before* their seconds are accepted. It is **PixInsight only**; under Siril the button is disabled.

1. Pick a scope. **One filter is the default**, because measurements are only comparable within a filter; *the whole project* still gives one distribution per filter. The dialog estimates the time — about 14 s per frame the first time, near-instant afterwards for frames already measured — and PixInsight is held for the whole pass. **Measure** starts it.
2. When measurement finishes, a **Frame selection** window opens with a histogram per measurement. Tick a measurement to set a limit and drag it across the distribution; a new limit starts at the edge and rejects nothing until you move it. The header shows how many frames the limits keep, and how many are *new since your last review*. **Reject frames whose star fit failed** applies to every filter. **Remember these limits for this project** stores them for later stacks.
3. **Reject frames outside these limits...** opens a confirmation naming each frame and its reason.

Rejecting moves the **original** light into the rig's ignored folder and sends its calibrated copy to the Recycle Bin — or, on a network share, which has no Recycle Bin, to the library's `.removed\` folder. Both are recoverable. A frame that could not be measured is never rejected.

Note: The Targets page count doesn't change when you reject. After the rescan, an accept proposal opens; apply it to correct the count.

Accepting lights

Accept lights N opens *Accepted subs from the lights on disk*. It counts each project's lights as the last scan saw them — `rejected\` folders, calibrated outputs and masters excluded — and matches them to your target's filter rows by filter and exposure (then gain and offset). Each row shows `old → new subs`, with decreases in red, and a checkbox. N is the number of *new* lights; a row that only goes down still makes the button worth opening.

- Nothing is written until **Apply N of M row(s)**, which sets accepted time to lights × exposure and marks those targets reviewed.
- A row with no lights on disk is left alone, never zeroed — unless its lights are found in the `rejected\` folder, in which case zero is proposed.
- Online-only files are skipped, so a partly downloaded folder under-counts; the dialog warns you.
- In a mosaic, lights carrying a name you merged into a panel (see [Stacking a project](#)) count toward that panel, whether or not their folders have been tidied yet.

Note: Moving a target on the Targets page zeroes its accepted counts, and the next Accept will propose the old counts again from the lights still on disk. Move the pre-move lights into `rejected\` if they no longer belong. See [Targets](#).

Renaming a project

Rename... on the card renames the project folder where it sits — nothing is copied and the frames do not move. It is what you want after merging a split mosaic back together, since the folder that survives is still named after one

panel.

A folder NightFront itself names after its target cannot be renamed here: the Rename field says so and points you at the target on the Targets page, since your next Save would rename the folder back to match. Folders you linked to a target by hand, and unlinked ones, rename freely.

NightFront records which frames you have reviewed and which have been stacked, a project's stacking settings, its target link, object-name rulings and unsent blink verdicts against the folder's hidden project id, not its name, so renaming a project in File Explorer loses none of it.

 **Rename here, not in Explorer — for a folder with no project id.** If NightFront could not write into a folder, it could not give it an id, and that folder's records are still filed under its name: renaming it outside NightFront loses all of that, and the project comes back reading as never reviewed and never stacked. The confirmation dialog says so when it applies.

Renaming onto a name another project already uses is refused — merge them instead. So is renaming while frames are still waiting in `Inbox\` or mid-import in `.intake\`, or while a job sent to a processing server is using the project.

Check layout... (projects from before the per-panel folders)

Older projects stored calibrated frames by filter only — `Calibrated\<filter>\`. For a mosaic that was unsafe: NINA restarts its frame numbering each session, so two panels' `LIGHT_0001` produced the same calibrated name, landed on the same file, and one silently replaced the other. NightFront recorded both as calibrated, so nothing anywhere showed it.

Check layout... appears on a project until the check has answered once. It reads each calibrated frame's own header to work out which panel it belongs to, and reports:

- **Nothing to do** — the project has no mosaic panels, or is already organized by panel.
- **Frames can be organized** — no two panels ever shared a name here, so the calibrated frames are simply moved into `Calibrated\<panel>\<filter>\`.
- **Frames were lost** — how many, per panel and filter. **Re-calibrate this project.** Your original lights were never touched, so re-calibrating rebuilds every lost frame *and* produces the new layout. Moving the survivors instead only fixes the folders; the lost frames stay lost.

Files whose `OBJECT` the project does not recognize are left exactly where they are and counted, not guessed at. Nothing is ever overwritten.

Tidying object-name folders

When you merge a misspelled name into a mosaic panel (see [Stacking a project](#)), the stack uses the ruling straight away, but the frames stay where they were filed: under `Originals\<old name>\` and `Calibrated\<old name>\`. While those folders exist, the project card says so, and new frames of that name keep going there too, so the folders stay consistent. The note appears after the next scan — applying the ruling starts one — and the stack that follows a new merge ends with a line pointing at it. This is not **Check layout...**, which answers once and then disappears.

Tidy folders... on the card lists every move before anything happens: per ruling, how many files move from which old folders into the panel's folders, and any ruling that cannot be tidied, with why. **Move N file(s)** then moves the old name's originals — its `rejected\` folder included, when it has one — and its calibrated copies into the panel's folders, and removes the emptied old folders. Afterwards new frames of that name are organized and calibrated straight into the panel's folders.

- **Nothing is re-calibrated** — each calibrated copy moves with its original, and your review and stacking records stay with the frames. No FITS header is edited.
- **Nothing is overwritten.** A file that appears at a destination before the move runs is renamed with a `_2` suffix instead.
- A ruling whose frames share a **file name** with frames already in the panel (NINA restarts its numbering each session) is refused and the files are named; the other rulings still move.
- A ruling that merges one panel into another is never acted on in the folders — only a name that is no panel is moved into a panel.
- The whole tidy is refused while frames are waiting in `Inbox\` or mid-import in `.intake\`; file those first. While another processing run is in progress the dialog says it is busy and the move stays unavailable.

Heads up: A frame-removal ledger written before the tidy still names the calibrated copies' old folders.

Stack again before removing frames from an earlier stack's ledger.

Stacking a project

Stack calibrates any uncalibrated lights first, then stacks — see the note under [The step buttons](#). What follows describes the stacking half; calibration itself behaves exactly as pressing **Calibrate N** would.

Stack asks where the masters go — every run, not remembered — and fills in `<project>\Stacked\` for you. You can change it: pointing a run at a new folder is how you keep an earlier iteration instead of writing over it. A folder the tree scans (`Originals\`, `Calibration\`, `Calibrated\` or the library) is refused, because masters there would be read back as lights or masters.

A stack registers and integrates every calibrated light of the project, grouped by target name and binning, producing one master per filter (a filter needs at least 2 frames). A mosaic's masters go into `Stacked\<panel>\`, one folder per panel; any other project's go straight into `Stacked\`, on either engine. Each master is *also* named for its group as well as its filter — `Spaghetti_Nebula_Panel_2_Ha_1x1.xisf` — so that a file still says which panel it is after you copy it out of this tree into WBPP or a share. Its settings come from the Settings sub-tab's **Stacking** chip. During the run:

- **Frame selection** (PixInsight only, off by default). With **Manual** mode, the stack pauses after measuring and shows the same Frame selection window as the review, with **Apply and continue** and **Cancel run**. With **Automatic**, it applies the limits stored for this project; a project you have never reviewed rejects nothing.
- **Other object names.** If the calibrated lights carry more than one `OBJECT` name, the run stops and asks, per name: **Stack together as "...", Keep separate** (the default) or **Skip these frames. Apply and stack** remembers the answer for this project; the stack edits none of your files. In a mosaic, the panels' own names are listed as panels, with their frame counts, and never asked about — a mosaic whose names are all panels stacks without stopping. Only a name that is no panel (a typo such as `Panle 2`) is asked about, and it is offered **Merge into** ▶ any panel of the mosaic instead of *Stack together*. A merge also decides which panel's

accepted time those frames count toward, and where new frames of that name are filed once the old folders are tidied — see [Tidying object-name folders](#).

Heads up: In Manual mode a stack waits for you indefinitely and blocks every other run until you answer. If you want to start a stack and walk away, use Automatic.

Heads up: Stacking again into the same destination replaces the previous masters of the same name. Choose a new folder if you want to keep an earlier iteration. Masters of *different* groups never collide, so a re-stack of a mosaic replaces each panel with its own new master.

Heads up: If a re-stack would use FEWER frames than a filter's master was built from — you deleted `Calibrated\` to free space and haven't recalibrated everything yet, for example — the run stops and asks, per filter: *"Ha 1x1 was stacked from 120 frames; 108 of them are not calibrated now, so this stack would use 12. Calibrate first, or tick it to stack anyway."* Leave it unticked and that filter's current master is left exactly as it is; tick it and the smaller stack replaces it anyway. It only asks about a filter whose new stack would actually be smaller, net of any frames you've since rejected.

Each run writes `stacking-log_<date-time>.md` next to the masters — settings, stages, and the frames that did not make it — so a re-stack never overwrites the previous log. When **Keep intermediate frames** is on, the status area reports their size with **Open folder** and **Remove them**.

The log's **Integration** section states, per filter (and per panel of a mosaic, with a per-filter subtotal), how many frames actually reached that master and their summed exposure — a fact about this one run, read from the frame ledger, never from the target card. **It is not the same number as the card's Accepted (h):** registration and frame selection can drop frames the card still counts as Accepted, and a re-stack may use a smaller subset than an earlier one did. A frame whose outcome can't be confirmed as integrated (a Siril best-percent selection, or a PixInsight run that rejected on read) is counted separately as "of uncertain outcome" — never folded into the hours and never read as zero — and the total is marked as a lower bound whenever any of those show up. The exception is a master whose own file records what went into it: Siril writes the stacked frame count and total exposure into every master, so a best-percent master's row reads those, marked (*from the master*), and they count toward the totals like any other row. PixInsight writes no such count, so its rows stay as described. An empty ledger or no declared master reads as unknown, not `0.0 h`.

Frames whose outcome the engine didn't report — the survivors of a Siril best-percent selection, or a PixInsight run that rejected on read — are listed under their own heading, **Frames whose outcome the engine did not report**, not under the frames that did not reach a master: many of them are in a master.

The AstroBin acquisition CSV

Each stack also writes `astrobin-acquisition_<name>_<date-time>.csv` next to the masters, ready for AstroBin's **Import CSV** in an image's Acquisition section — instead of typing one row per night and filter. The date-time matches the run's stacking log.

- **One row per night, filter and sub length**, with the number of subs that went into the master. A night is the evening it started, so subs taken after midnight stay on that evening's row. Binning, gain, sensor cooling (`SET-TEMP`) and the rig's f-ratio are added when the frames and the [Imaging System](#) state them; a sub that differs in any of them gets its own row.

- **The filter column** is each filter's **AstroBin filter** number from Imaging System → Filters. A filter without one is still listed — count and sub length still import — with its filter left blank, and the stacking messages name the filter to fill in. OSC frames (no `FILTER`) use your one **Osc**-role filter's number.
- **One file per target** (per panel for a mosaic), beside that target's masters. A mosaic also gets `..._all-panels_...csv` in the stack folder, covering every panel.
- **What is counted** is the subs that reached a master — rejected and failed frames are left out. Where the engine can't say exactly which subs it kept (Siril's keep-best percentage, or PixInsight integrating slightly fewer than it was given), the count is the most it could be, and the stacking messages say so.
- The f-ratio comes from the Imaging System page as it is now, not from the frames — if you have changed reducers since some of the subs were taken, check that column before importing.

The file is only an extra output: nothing about how the stack runs changes, and if it can't be written the stack still succeeds and the messages say why.

The rejected-frame ledger

After a stack, **Rejected-frame ledger** appears in the status area, collapsed, with a summary such as *147 of 171 frames reached a master (2 rejected, 1 failed)*. Counts are of the calibrated lights the stack ran against. Expanded, it lists per group and filter the frames that did **not** reach a master:

- **Verdict** — *rejected* (by frame-selection limits) or *failed* at a named stage, or *unknown* when the engine could not say. *Unconfirmed* frames — PixInsight reported fewer frames integrated but not which — are counted but not listed.
- One column per measurement with its value and **z**, the robust distance from that filter's median. For FWHM, eccentricity and median, negative is better; for stars and PSF signal weight, positive is better.

Remove... on a row, or **Remove all N from ...**, is offered only for *rejected* and *failed* frames. The confirmation names the files. The original light moves to the rig's ignored folder and the calibrated copy goes to the Recycle Bin (or the library's `.removed\` on a network share) — both recoverable. The row stays, marked *removed*, and the rescan then opens an accept proposal to correct the count.

Completing a project

Complete archives a finished project and stops planning its target. It fills only once the project has stacked masters, nothing earlier has work, **and the linked target's collection goal is met** — re-stacking partway through a target's plan doesn't light it up early, and a folder that maps to no target never fills it on its own (you can still complete it by hand). It opens a dialog; nothing happens until you confirm.

- The folder moves into the rig's completed-projects folder (always `Completed\` in a managed rig; `Completed\` by default in one laid out the old way) and is no longer scanned. Its target — a mosaic as a whole — stops being planned, but stays in the config.
- If you open the dialog yourself before the goal is met, it says so — for example *"M 31 still has 4.2 h to collect (Ha 2.0 h, OIII 2.2 h)"* — as a statement, not a gate: ticking nothing still archives the project.

- A table headed **Tick to DELETE** lists the project's folders with sizes, a **Delete** column and a **Zip** column. Only `Calibrated\` and `Originals\rejected` are pre-ticked; `Originals\` and `Inbox\` are marked *Irreplaceable acquisitions*, `Calibration\` is rebuildable only while the library still holds those masters. Ticking a folder takes everything inside it. Warnings appear if your selection would leave nothing to regenerate from.
- Leftover calibrated copies and working-drive files belonging to this project are swept too, and listed under *Outside the project folder*.
- **Zip** compresses a kept folder, with an estimated saving and time; the archive is verified before the folder is removed. It can take over an hour for a large `Originals\` over a network share.
- The button reads **Archive only**, **Archive and zip**, or **Complete and delete X GB**.

Heads up: This is the only place NightFront permanently deletes data. Everything you tick is deleted — not sent to the Recycle Bin, and not recoverable from within NightFront. Completing with nothing ticked only archives.

A project can't be completed while frames are still waiting in `Inbox\` (run Organize first), or when the rig has no completed-projects folder set. A record of what was done, deleted and left alone is written into the archived folder as `.nightfront-completed.txt`.

To undo, select the **Completed** chip at the top of the Projects list and press **Restore**: the folder moves back (in a managed rig, into `Active\` or `Planned\` to match its target), is scanned again and its target is planned again. Restore waits for any run in progress, and it brings back the annotations and hand-picked masters NightFront keeps about the project's own master frames, pointing at the restored folder. The confirmation shows the completion record, because restoring cannot bring back anything that was deleted.

The Targets page's **Plan again** clears the completed mark too, as an edit you save like any other (**Save** highlights). In a managed rig it then moves the folder back out of `Completed\` once the config is saved, and lists the move in the **Folder changes** note; if the folder can't move yet (a file held open, a job on a processing server), it is tried again at the next open or save, or you can press **Restore** yourself. A project completed before you switched to the managed layout has no link to its target yet; **Plan again** still brings it back when it is the only completed folder with exactly the target's name, and links it. If more than one could be it, nothing moves and the note asks you to **Restore** the right one. In a rig laid out the old way, **Plan again** does not move the folder.

Library sub-tab

The Library lists every master the rig knows, grouped the way matching thinks about them: **Bias**, **Dark** and **Darkflat** masters by camera settings (dark and dark-flat groups list the exposures they cover), **Flat** masters by night (and rotator angle, when the rig matches flats on rotation), then settings, then filter. Groups start collapsed. Each master shows which light groups use it and its key fields; a value marked (*EGAIN*), (*name*), (*annotated*) or (*rig default*) was recovered from somewhere other than the header.

- ← **calibration uses this one** marks which of several same-key bias or dark masters calibration takes. When a master exists as both `.fit` and `.xisf`, PixInsight uses the `.xisf` and Siril the `.fit`.
- ▲ **Move to library** moves a project-local or external master into the library (the file leaves where it was).
- **Remove / Remove duplicate** discards a project-local or external copy (Recycle Bin). **Remove all N duplicate(s)** does it in one pass, only for copies proven byte-identical to one that stays.

- **Remove from library...** removes the library's own copy. The dialog lists the light groups that use it and whether any other copy exists.
- **Remove** followed by the night's date, on a flat night's header, removes every master flat of that night, library copies included.
- **Annotate...** lets you type key fields a master never recorded (a WBPP master often has no temperature, for example). Blank means *not stated*. *unresolved key* in red means the master can't match anything until a field is stated. Saving rescans.

Heads up: The library is the reference every project of the rig calibrates against. Removing a library master can leave light groups with a gap on the next scan. It goes to the Recycle Bin, or the library's `.removed\` folder on a share.

Archiving replaced masters

Flats are matched to each light by date, and Organize copies the flats a project uses into its own `Calibration\` folder, so an older master flat that a newer one has replaced usually has nothing left to do. After every scan NightFront checks the library for such masters (flats, and also darks, biases and dark-flats that a newer master of the same settings has replaced) and, when it finds some, shows one line on the Library page: *"7 master flats and 2 darks have been replaced by newer ones and are no longer needed."* Nothing moves on its own.

Click **Review...** to see them, grouped by kind and by night, each with its reason (for example *"replaced by 2026-09-14; 3 projects hold a working copy"*). They start ticked; **Archive N masters** moves the ticked ones. A closed **Kept** group lists the replaced masters that must stay, and why: a project that uses one has no copy of it yet (organize that project first), a copy lacks a setting the original had, a hand-picked master or pre-calibration master points at it, it is the master PixInsight would use, or some of a project's lights could not be read. NightFront checks everything again at the click and moves only what is still safe.

Archiving is a move, never a delete. Each master goes to `Calibration Library\Archive\Masters\`, keeping its place in the library's folders, and a line for it is added to `archive-log.txt` there. The **Archived masters** section at the foot of the Library lists them by kind and date; **Restore** moves one back to exactly where it was (refused if something else now sits there). NightFront never deletes from `Archive\Masters\`; clear it out yourself in Explorer when you are sure.

Heads up: If you turn on **Match flats on rotator angle** after archiving, a flat you now need may be in the archive; **Restore** it. And once a project's master is archived, the project's own copy is the only one left, so keep it.

Building masters from calibration subs

Build masters from calibration subs is a collapsed panel at the top of the Library.

1. **Scan for subs** finds bias, dark, flat and dark-flat subs under every project's `Calibration\`, plus any folder you add with **Add folder...**
2. The plan lists one master per key, with checkboxes, **Select all / Select none**, and the reason a group is blocked (for example, a master of that name is already in the library). Flats are pre-calibrated with a matching master bias; **Choose...** picks one by hand when temperatures disagree.

3. **Show script** displays the generated script. **Build N master(s)** runs the engine.

Masters land in the library with their full key written into the header. The subs of each master flat move to `Calibration Library\Archive\Flats_<night>\`, listed under **Archived flat subs** with **Remove archived subs**. After a build, the tree is rescanned and every project that now uses a new master is organized automatically.

To leave flat subs where they are instead, tick **Keep flat subs where they are when masters are built** on the Processing Settings sub-tab's **Import chip** (off by default). Every archive step then puts a verified copy of each sub, and of each rejected flat, into the same `Archive\Flats_<night>\` set, and the originals stay exactly where NINA left them, for you to delete yourself. A copy counts only once it has been read back and found identical. A sub already in the archive under the same name and size is left alone, so archiving again copies only what is new, and a group whose subs are all in the archive is not offered for archiving again.

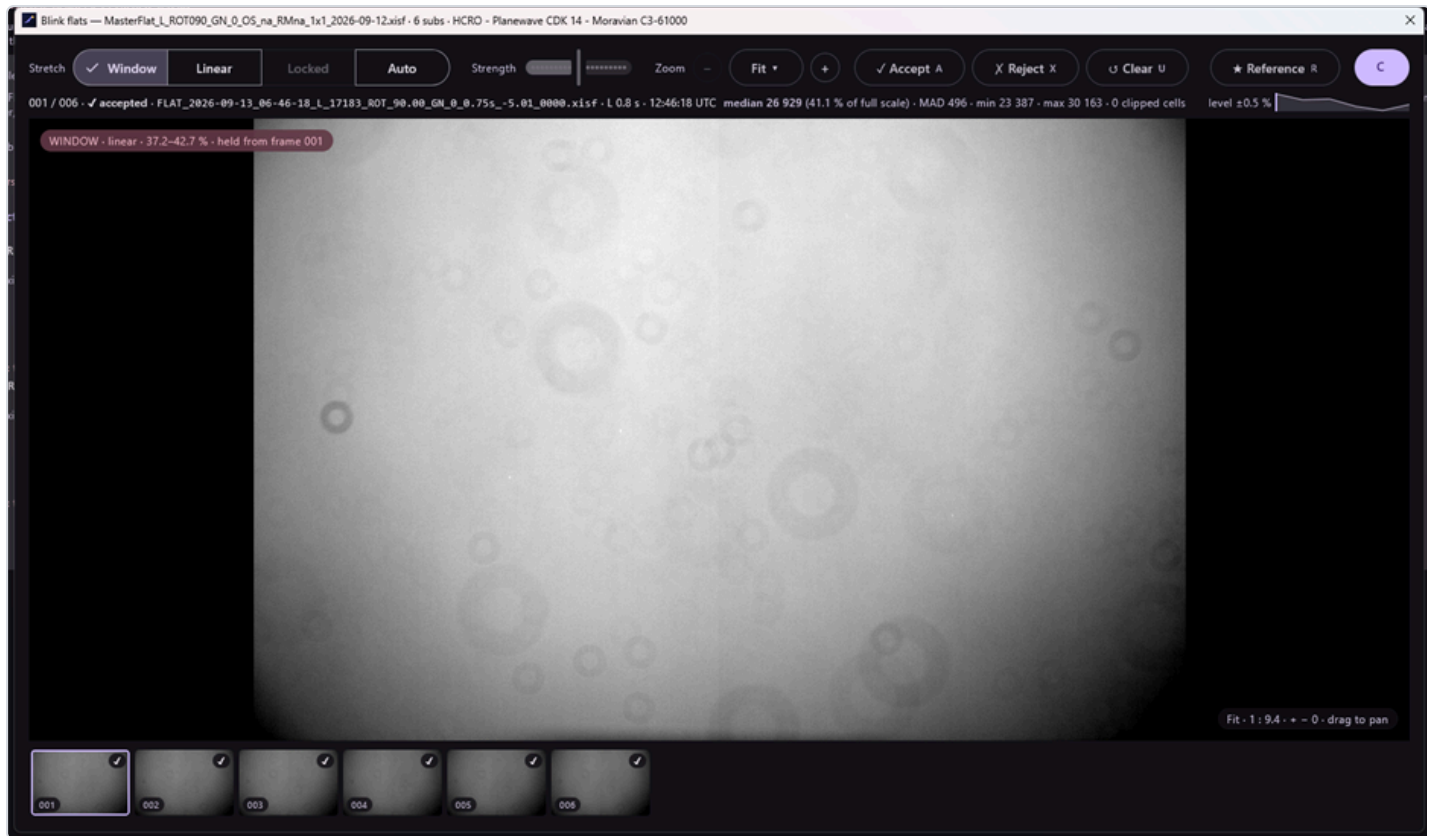
Heads up: The automatic organize after a build moves lights in those projects, just as pressing **Organize** would. **Remove archived subs** is permanent on a share with no Recycle Bin.

Archive the subs of N already-built group(s) files away subs whose masters are already in the library.

Blinking flat subs

A flat can be wrong as a whole — a slipped dew shield, a flickering panel, a sky flat taken as the sky darkened — and the integration's pixel rejection cannot remove it. Blinking a flat group first lets you reject such a flat before it reaches the master. It is optional and never blocks a build.

- **Blink** on a flat row opens that group. A row blocked as already in the library has none; its subs are done, and **Archive these subs** is the action there.
- **Build N master(s)**, when ticked flat groups have not been blinked this session, first asks *Blink the flat subs first?* with the number of groups and subs, and three buttons: **Build without blinking** · **Cancel** · **Blink first**. **Blink first** opens the groups one window at a time and stops there; press **Build** again when you are done. **Don't ask again** turns the offer off; turn it back on under [App settings](#) > Paths.



The window is the one described in [Blinking a night](#), with the same keys, and every flat opens accepted. Three things differ for flats:

- **Stretch** opens on **Window**: a straight ramp between two levels taken from the first frame (or the reference, once you set one) and held for the whole group. Vignetting and dust show plainly, and a brighter or dimmer flat looks brighter or dimmer. **Linear** and **Auto** are one **S** press away (hover over a button to see what it does); **Strength** has no effect in Window or Linear and is disabled. There is no **Locked** for flats: Window already holds one frame's levels for the whole group.
- The curve is labeled *level*, with the group's spread as a percentage (for example *level ±0.5 %*). A flat whose median is 3 % or more off the rest of the group is marked for a look — often the first one to three flats of a sky-flat sequence, while NINA's automatic exposure settles.
- The status line adds the median as a percentage of the file's full scale.

Enter with nothing rejected goes straight to the next group. With rejects, a confirmation lists where they go and what the change does to the plan — a group that would drop below the minimum, be renamed, or lose a pre-calibration master you chose. **Rejected flats move into the rig's ignored folder** (`rejected\` by default) beside the others; **accepted flats never move, and stay in the group to be integrated**. **Cancel** reopens the window with your verdicts intact. **Esc** stops the queue; groups already committed stay committed.

While a flat blink is open, Build, Scan for subs, Add folder..., Choose..., the Archive buttons and the calibration engine switch are unavailable. When the queue ends, the plan is re-made and your un-ticked groups stay un-ticked. "Blinked this session" beside a row is forgotten when the app closes.

When a master flat builds and its source folder holds no more flats, the folder's `rejected\` moves with it into `Calibration Library\Archive\Flats_<night>\rejected\`, beside the rest of the set. With **Keep flat subs where**

they are when masters are built ticked, the rejected flats are copied there instead, and the folder's own `rejected\` stays as it is.

Note: A flat that is not rejected is integrated — there is no way to hold a flat back other than rejecting it.

Processing Settings sub-tab

The Settings sub-tab holds what is particular to **this imaging system and its rig folder**, as four chips: **Rig**, **Import**, **Calibration** and **Stacking**. A line under the chips says whose settings the selected chip edits. Folder paths are chosen with **Browse...** and removed with **Clear**.

Settings that belong to the whole computer — the calibration engine, the PixInsight and Siril executables, the Projects root, the working folder, where processing runs, the processing server and notifications — are not here. They are in **App settings**, the button in the title bar (see [App settings](#) below), because they are the same whichever config is open.

Copy settings from..., beside the chips, copies the **Calibration** and **Stacking** settings (master flat integration, cosmetic correction and stacking) from another imaging system. It copies the *recipe*: a defect-map path is machine-local and is never copied, and the **Import** chip's download folder is never copied either, because each rig's lights arrive in their own folder, and nor are that chip's two ticks.

The Import, Calibration and Stacking chips are stored per imaging system and need the config to have an imaging-system name; without one the page says so and hides them. The Rig chip works regardless.

The Rig chip

The chip starts from one **Rig folder** picker for the imaging system. It lists the folders serving this config and chooses the primary one; **Rescan** sits on the picker's row, and **Rename...** and **New rig folder...** sit beside it. Under it a *Layout* line says whether the rig is managed or laid out the old way, and for the latter **Adopt managed layout...** sits under its explanatory sentence. The rig's other settings follow; **Advanced** (collapsed) holds the ignored folder names and, only for a rig laid out the old way, the completed folder's name; **Other rig folders** (collapsed, with a summary of what it holds) lists every other folder under the Projects root with a box to claim it for this config and, on a claimed one, **Make primary**. These settings are written into the rig folder's own `nightfront-processing.json`, so they travel with the data.

Adopt managed layout... first shows a preview of every project and where it will go: **Active** or **Planned** to match its target, **Active** for a project with no target, and the contents of the old completed-projects folder into **Completed**. A folder whose name differs from its target's has a **rename to match** box, unticked: ticked, the folder is renamed to match and follows later renames of the target; unticked, it moves and keeps its name for good. Nothing is copied or deleted, and a name already taken in `Completed\` refuses just that folder.

Setting	What it does
Ignored folder names in ...	Comma-separated; any folder with one of these names inside a project is skipped. Default <code>rejected</code> . Rejecting, removing and blinking need at least one.
Completed projects folder in ...	Only for a rig laid out the old way, under Advanced : where Complete moves a project. Default <code>Completed</code> . Empty means projects can't be completed. A managed rig always uses <code>Completed\</code> , and the field is replaced by a line saying so.
Default offset / Default readout mode	Used for a master whose header, name and annotation say nothing.
Readout mode aliases in ...	One <code>alias = canonical</code> per line, for one camera mode spelled two ways. <i>Seen on this rig</i> lists the spellings the last scan read.
Match flats on rotator angle ($\pm 1^\circ$)	On by default. Off means one flat serves every angle.

Note: The rig-folder text fields are committed with **Apply**. The ignored-folder, completed-folder and alias fields say *Not applied yet* in red until you do; the two folder-name fields also apply on Enter. Applying rescans the tree.

Note: A refused rig-level action (Rename..., New rig folder..., choosing the primary folder, claiming a folder or Adopt managed layout... — for example while a job sent to another machine holds the rig) says so directly under the **Rig folder** row. An imaging system that has never been given a name shows "...serve this untitled imaging system — name it on the Imaging System page..." in its scan line, and the Rig chip explains that no rig folder can be linked until it has a name.

Note: Rig folders are filed by the hidden `.nightfront-rig-id` NightFront writes into each one, so **Rename...** (or renaming in File Explorer) keeps everything. **Rename...** is refused, with the reason, while a run is in progress, while a job sent to a processing server is using the rig, when the new name is taken, or for a folder that has not been given its id yet (usually because its `nightfront-processing.json` could not be read — fix that, **Rescan**, and try again).

The Import chip

Setting	What it does
Download folder	Where this rig's lights arrive; needed for Import.
Delete each file from the download folder after importing it	Off by default. See the warning under Importing lights .
Keep flat subs where they are when masters are built	Off by default: flat subs move into the library's <code>Archive\</code> . On: a verified copy goes there and the subs stay where NINA left them. See Building masters from calibration subs .

The Calibration chip

Setting	What it does
Master flat integration	Rejection algorithm and parameters for building master flats, matching WBPP's flats panel. Auto picks by sub count. Reset to WBPP defaults .
Image Calibration	What Calibrate reads, as Stacking is what Stack reads. One collapsible group, Cosmetic correction , holding ImageCalibration (built-in hot/cold sigma) and CosmeticCorrection process (PixInsight only: auto-detect hot/cold, defect map). See Cosmetic correction below. Reset image calibration to defaults .

The Stacking chip

Setting	What it does
Stacking	Collapsible groups: Registration , Frame selection , Local Normalization (PixInsight) or Background Extraction (Siril), Integration , Drizzle , Output . Reset stacking to defaults .

Notable stacking options (all on the Stacking chip):

- **Frame selection** — **Narrow each stack to the frames that pass this project's limits**, then **Manual** — **review the histograms mid-run** or **Automatic** — **apply this project's stored limits**.
- **Drizzle** — changes how color (OSC) lights are calibrated, so an OSC project's calibrated frames are rebuilt when you turn it on or off.
- **Keep intermediate frames** — keeps the registered frames after a successful run. A failed run always keeps them.
- **Add stack date and integration time to master names** — off by default. When ticked, every master you stack ends with the day it was stacked and that filter's integration time (the frames actually stacked, to the nearest minute), for example `Spaghetti_Nebula_Panel_2_Ha_1x1_2026-10-04_12h30m.xisf`; the crop and drizzle versions follow the same name. A re-stack on another day, or one whose integration time differs by at least a minute, leaves the earlier master beside the new one for you to delete; a same-day re-stack that rounds to the same integration time gets the same name and replaces it. If a run cannot establish the integration time, the name carries the date only.
- **Plate-solve the master** — Siril only; runs after the masters land and needs network access to a star catalog. A failed solve never costs the master.

Cosmetic correction

The **Image Calibration** section's group, on the Calibration chip. How each calibrated light's defective pixels are corrected — hot, cold and (PixInsight only) a named defect list. There are two groups, and each pixel is corrected in exactly one of them, never both:

- **ImageCalibration (built-in, needs a master dark)** — the sigma test against the master dark that calibration has always run: **Hot** ☒ σ (on by default, $\sigma 10$) and **Cold** ☐ σ (off by default). Runs on both engines. A job with no master dark (a bias-only calibration) skips both, whatever is ticked here.
- **CosmeticCorrection process (PixInsight only)** — **Auto-detect hot** ☐ σ and **Auto-detect cold** ☐ σ run PixInsight's own `CosmeticCorrection` process instead of the built-in pass for that axis; ticking one grays the matching row above with *handled by CosmeticCorrection*. Under Siril this whole group is grayed, and the

header says so: an auto-detect box you leave ticked still takes effect on your next PixInsight run, it just falls back to the built-in pass while Siril is selected.

- **Defect map** — a file picker for a linear defect map: the column/row list PixInsight's own *LinearDefectDetection* script writes. **Clear** removes it. Set the **binning** it was measured at (default 1x1); a project calibrated at a different binning, or at a binning NightFront can't determine, skips the map for that job and says so, without touching its hot/cold correction.
- **Amount** blends how strongly a corrected pixel replaces the original (1 = fully replaced).
- The status line under the file picker parses the map as soon as you choose it and reports rows found, or the problem. **A map containing a row defect is refused, whatever the binning** — PixInsight's *CosmeticCorrection* writes zeros into part of every row defect on some frames and still reports success, so NightFront won't run one. Point the map at *LinearDefectDetection*'s detected-**columns** file instead, or remove the row lines; column defects are unaffected.

Leaving every option in this section at its default reproduces exactly what calibration has always done. Copy settings from... carries the whole recipe except the defect-map path (see [Processing Settings sub-tab](#)); a recipe change does not re-calibrate lights already recorded as calibrated — delete a light's calibrated file to have it redone under the new recipe.

App settings

The  **App settings** button in the title bar, beside your open configs, opens a window with three chips: **Paths**, **Server** and **Notifications**. Everything in it belongs to this computer: it applies to every open config, is stored in your user profile, never in a config, and the NINA plugin never reads it. It needs no config open, and each change takes effect as you make it. A refusal is shown beside the control that caused it.

The **Paths** chip:

Setting	What it does
Calibration engine	PixInsight or Siril . Switching rescans; stored settings are never changed. Can't be switched while a flat blink is open.
Run processing on:	Local (this machine) , or a peer you've added. With a peer picked, Calibrate, Build masters and Stack all run there instead. Manage peers... adds and removes them. See Remote processing .
Offer to blink flat subs before building master flats	On by default. Pressing Build first offers to blink ticked flat groups not yet blinked this session. See Blinking flat subs .
PixInsight executable	Required for Calibrate, Build and Stack under PixInsight, and for Review.
Siril executable (siril-cli.exe)	Required for Calibrate, Build and Stack under Siril.
Projects root	The folder holding one sub-folder per imaging system. Choosing one scans it.
Working folder (intermediate frames)	Scripts, logs and intermediate frames during a run. Must be a local drive; a stack can need tens of GB. Defaults to <code>.config\nightfront\runs</code> in your user folder, on the system drive.
Clean working folder...	Removes intermediate frames from every working folder this machine has used. Scripts, logs and run results are kept; frames you asked to keep are removed too. Unavailable during a run.

The **Server** chip is where this machine offers itself as a processing server; see [Remote processing](#). The **Notifications** chip sends a Telegram or Pushover message when a step you choose ends; see [Notifications](#).

PixInsight and Siril

Both engines do the same jobs — build masters, calibrate, stack — but not every option exists in both. A setting one engine can't express is grayed and labeled in that engine's view, and the collapsed group headers say what *this* engine will actually do. **Switching engines never changes a stored setting**; switch back and your choices are still there.

Capability	PixInsight	Siril
Review before accepting, and frame selection during a stack	Yes	No
Plate-solve the master	No	Yes
Local normalization	Yes	No (subtracts each frame's background instead)
Averaged sigma clipping and RCR rejection; large-scale pixel rejection	Yes	No
Autocrop	Yes	No
<i>Common area only</i> framing; point, turbo, lanczos2/3 drizzle kernels	No	Yes
PSF signal weighting	Yes (default)	No — the run weights frames equally and says so
CosmeticCorrection auto-detect and defect map	Yes	No — falls back to the built-in hot/cold sigma pass

Siril reads `.fit` masters. When a master exists only as `.xisf`, NightFront converts it before the run and saves the `.fit` beside the `.xisf`: with Siril itself if your Siril build reads XISF, otherwise with PixInsight if its executable is set. If neither can, the run refuses before starting and names the masters that need a `.fit` copy.

Online-only files

If your Projects root is a cloud-synced folder with on-demand files, reading even a header can download the whole frame. NightFront checks before touching a file: online-only files are **skipped and counted, never fetched**, and the status area says how many, suggesting you point NightFront at the NAS's network path or pin the folder to keep it on the device. Review and Accept skip them too, so counts can be low until the files are local.

A network path keeps frames off your disk; a local copy is much faster (about 0.2 s per frame against 0.8–3.2 s over the network), which is usually worth it for a project you're actively working on.

Remote processing

If the fastest machine on your LAN isn't the one running NightFront, it doesn't have to be. Any NightFront can offer itself as a **processing server**; any other can send it a Calibrate, Build masters or Stack job to run there instead of locally. The result comes back and is recorded on your machine exactly as a local run would be — same run record, same ledger, same accepted counts.

This is a general capability, not a fixed pairing of two machines. You choose which machine to send work to; NightFront shows you what each one offers (its engines, whether it can reach your Projects folder, how much CPU and free disk it has) but never recommends one — it has no way to know which machine is actually faster for your setup.

Tested on Windows and Linux, in both directions. NightFront builds and ships on macOS too, and its own tests run there, but processing across two machines with one of them running macOS hasn't been tried yet.

Setting up a server

On the machine you want to do the work — call it the server — open **App settings** in the title bar, next to your open configs, and choose its **Server** chip. The window needs no config file open at all: it is a setting of the machine, not of any project.

1. Set **PixInsight executable**, **Siril executable (siril-cli.exe)** and **Projects root** on the **Paths** chip of the same window. The Projects root is *this machine's own path* to the tree; it can be spelled completely differently from how the machine that dispatches to it sees the same folder (a drive letter here, a network path there).
2. Set the **workspace root** — scratch space for a job's intermediate files. Unlike the Projects root, this should be a **local disk**, not a network share: PixInsight is known to fail silently when its working folder is a network path.
3. Choose a **bind address** (the address a peer will type in to reach this machine) and a **port** (51820 unless you have a reason to change it), then turn on **Serve processing jobs**. A token is generated the first time you do this — **Copy** it; every machine that should be able to send work here needs it, like a password. **New token** invalidates every peer configured with the old one.
4. **Serve at launch** starts serving automatically the next time NightFront opens on this machine — the only way to reach a server started from macOS Login Items, which can't be given a command-line flag.

The first time you turn on Serve, your operating system's firewall will very likely ask whether to allow NightFront to accept connections. Allow it — otherwise a peer will report this machine as unreachable with nothing telling you why.

Turning Serve off does not stop a job already running. It only stops accepting new ones; a job in progress keeps going and stays reachable again the moment you turn Serve back on. Closing every config tab on the server doesn't stop a running job either — the server lives at the app level, not inside any one tab. If you close the whole app while a job is running, NightFront asks first, naming the job; if you let it proceed, the job is cancelled cleanly. An unplanned shutdown (a power loss, an OS update forcing a restart) is recorded as *interrupted* rather than lost — the next launch on this machine, or the next time the dispatching machine asks, reports it that way.

The server holds no config, no targets, no rig — nothing beyond what's above. It can't see anything about your imaging plan; the job it's handed carries everything it needs to run.

Adding a peer

On the machine you'll dispatch *from*, open **App settings** and its **Paths** chip. Beside the calibration engine choice is **Run processing on:**, defaulting to **Local (this machine)**. **Manage peers...** opens the list of machines you can send work to.

To add one, enter the name you want to call it, and the address, port and token from that machine's own App settings › Server chip, then press **Check**. NightFront fetches what that machine reports about itself — its app version, which engines it has and their versions, whether its Projects path is readable and writable, its CPU count and free disk space — and shows it before you save. **Save peer** stays disabled until a Check against exactly those values succeeds; editing any field afterward clears the check, so you can never save a report that doesn't match what you typed.

The peer picker always reopens on Local when you relaunch NightFront. It never remembers a remote selection between sessions, so a new session is never silently aimed at a machine that might not even be online.

What runs remotely

With a peer picked, **Calibrate**, **Build masters** and **Stack** all send their job there instead of running here — you don't need PixInsight or Siril installed on this machine at all if every project you work on has a server to send jobs to. **Review always runs locally**: it measures lights already on disk and has no engine work to hand off.

A machine with no local engine configured can still Calibrate, Build masters and Stack — the button is enabled as long as *either* a local engine or a picked peer can do the work.

Stack calibrates first on the peer too: any uncalibrated lights are calibrated there, then stacked there, as one action. If you change **Run processing on:** while that calibration is running, nothing is stacked and the report says so — press **Stack** again to stack where it now points.

The **Active projects** box's **Calibrate** and **Stack** send one project at a time to the picked peer, waiting for each job to come back before sending the next. If this tab is still following an earlier job on a peer when you start one, the batch waits for that job too rather than skipping every project.

Sending to a peer doesn't need this machine's engine, so **Calibrate** in the *Active projects* box keeps sending projects while this tab runs something else locally — a Review's measurement, a local Calibrate, a master build for another rig — and you can start it while that is running. It waits only when the local run is working on the very project it would send next, is building masters for that project's rig, or is a Siril calibration on that project's rig (Siril adds FITS copies of the rig's masters to its master library before it calibrates, and the peer would otherwise be making the same copies at the same moment). A FITS copy only ever appears in the library once it is complete, so a Siril calibration — here or on a peer — never reads one that is still being copied. It also waits the few seconds a **Stack** for the peer takes to be planned on this machine, since this tab follows one job on a peer at a time. Its **Cancel** stops the batch and its job on the peer, never your local run. **Stack** in the *Active projects* box does the same for a project whose lights still need calibrating: that calibration is sent to the peer while your local run goes on, and

you can start Stack while the local run is going. Each project's stack is planned on this machine before it is sent, though, so that part waits until the local run has finished (the project's panel says it is waiting), and a project with nothing to calibrate waits for it before anything is sent. The **Stack** button on a single project still waits for local runs to finish.

While a job is away

Calibrating or stacking holds the **project** it's working on; building masters holds the **whole rig** — its master library and its flat/dark/bias subs — because a masters build can touch any project's masters on that rig. Anything on this machine that would otherwise move, rename or delete files the remote job is reading or writing is refused for as long as the job is away, naming the action, the job and the peer it was sent to, with **Abandon remote job...** as the way out if the peer is gone for good.

The hold is also visible to other computers. While a job is away it leaves a small hidden `.nightfront-remote-job` file in the project folder (or in the rig folder, for a masters build), and another computer that reaches the same Projects folder refuses to rename, merge, complete, restore or move that project between Active and Planned until the file is gone. It is removed when the job is collected or abandoned; delete it by hand only if the computer that sent the job is gone for good. A job is not sent at all when the file can't be written.

Other rigs, and other projects during a Calibrate or Stack, stay fully usable. Only a masters build reaches beyond its own project — stated plainly in its refusal, not something you discover by trying.

Collecting the result

While the job's window is open, you see it running here just as a local run would show, including a live Cancel. If you close NightFront, or lose the network, while a job is away, nothing is lost: the next time NightFront opens — or reconnects — it asks every peer it knows about for jobs it's still waiting on, collects anything that finished, and reports it ("*Stack of M 31 finished on astroprocess at 03:12 — ingested*"). A job the peer no longer knows about (a reinstall, a different machine now at that address) is reported as such rather than retried forever; **Abandon remote job...** is the way out there too, and it tells you which lights will be re-offered for calibration once you do.

What's different about a remote job today

- **Manual frame selection is refused for a remote stack**, never silently run automatically instead. If your stacking settings use manual frame selection (the default), switch to automatic selection before sending a stack to a peer, or stack locally. The one exception is the **Active projects** box's **Stack**, which runs frame selection automatically for that run wherever it runs — its confirmation says so before you start.
- **A stack or masters build collected after NightFront was closed for the whole run** gets a plainer record than one you watched live: the stacking log has no narrative section (the AstroBin CSV is still written), and a collected masters build doesn't automatically archive the flat subs it used — press **Archive all** on the next scan to do that by hand.
- **One job at a time per server.** A job sent to a peer that is already busy — with another job, or with a run of its own — is refused, saying what the peer is busy with; it isn't queued. Send it again once the peer is free.

The **Active projects** box's **Calibrate** and **Stack** don't give up like that: they wait for the peer to finish, then send each project in turn — their status line says what the peer is busy with, and **Cancel** ends the wait.

- **Cancel shows what it did, at once.** Pressing Cancel on a remote row shows "*Cancelling on astroprocess...*" immediately and grays the button. If the peer's engine is still running when the Cancel arrives, it stops there and nothing is kept — the run reads "*cancelled on astroprocess*". If the Cancel arrives after the engine has already finished — the peer is writing the result back into your tree — it can't be stopped any more: the result is kept, and the summary adds "*Cancel arrived after the engine on astroprocess had finished; the result was kept*". Either way, a pressed Cancel never lets a Calibrate → Stack chain go on to stack what was calibrated.
- **The server needs a real desktop session** — it's a mode of the same app you're running, not a background service, so the server machine needs to be logged in and NightFront running (or set to serve at launch).

See [Processing](#) for calibration, review, stacking and the rest of the closed loop this sits inside of, and [Troubleshooting](#) if a peer won't respond.

Notifications

NightFront can send a message to your phone when a processing step ends, so you can start a long stack, walk away, and find out how it went without watching the screen. Messages go through **Telegram**, **Pushover**, or both, using a bot or application that you create yourself. NightFront does not run a notification service: the messages are sent from your own machine, by your own bot or application.

Everything is set up in **App settings** (the button in the title bar, next to your open configs), on its **Notifications** chip. The settings belong to you and this computer: they are stored in your user profile, never in the config file, so they do not travel with a config to another machine and the NINA plugin never reads them.

What sends a message

Turn on **Notify me when processing steps end**, then tick the steps you want to hear about. The **All** and **None** buttons tick or clear the whole list. The first time you turn notifications on, every step is ticked, so the first thing you run tells you it works.

Step	Sends a message when
Calibrate	a calibration run ends
Build masters	a masters build ends
Stack (including calibrate then stack)	a stack ends. Calibrate-then-stack is one message, sent when the whole chain ends, not one for each half
Review measurement	a review measurement ends
Batch (Calibrate all / Stack all)	the whole batch ends: one message listing every project, not one per project
Import	an import ends
Complete project	a completion ends
Remote job finished	a job you sent to a processing server is recorded as finished on your machine (see below)
Needs your input	a step stops because it needs a decision from you

A step you leave unticked sends nothing. With the master switch off, nothing is ever sent.

Each message says which step ended, how it ended, the project and rig, the run's own summary sentence, which machine ran it and how long it took. The title reads like `NightFront • Stack finished – M 42 (CDK14)`. A step ends one of six ways, and the title says which: **finished**, **finished with errors** (some stages failed), **failed**, **cancelled** (you pressed Cancel), **could not start** (refused before it began, for example because the processing server was busy), or **needs your input**.

Needs your input covers a step that stops and waits for you: the stack's manual frame-selection review, the question of which object name a project belongs to, thin masters that need your OK, an import that proposes a routing for you to confirm, and a review measurement that opens the review. These messages are sent under

Needs your input only, whichever step stopped: with Stack ticked and Needs your input unticked, a stack that stops to ask you something sends nothing (and never a "Stack finished" for a stack that did not run).

Deliberately not offered: the flat-blink queue and its commit, Blink commit, Archive, Copy to library, Remove frames, Organize, Clean working folder, Rename and Merge. These are quick and interactive, not walk-away work. The nightly and seasonal planning runs do not send messages yet.

Closing a tab, or closing NightFront, while something runs is not an ending and sends nothing, so quitting the app does not send a burst of "cancelled" messages. The one exception is a job running on a processing server whose result is still brought back after its tab closed (see **Limits worth knowing** below).

Setting up Telegram

1. In Telegram, open a chat with **@BotFather** and send `/newbot`. Give the bot a name and a username (the username must end in `bot`). BotFather replies with a **bot token**, a long string like `123456789:AA...`. Treat it like a password.
2. In NightFront, paste the token into **Bot token (from @BotFather)**. The field is masked; **Show** reveals it.
3. Press **Link...** NightFront checks the token and shows your bot's name. Now, in Telegram, open your new bot and press **Start** (or send `/start`). A bot can only message people who have messaged it first, so this step cannot be skipped.
4. NightFront picks the message up (it listens for up to two minutes, and only while the dialog is open) and shows the name of the person or group that sent it. If it is you, press **Link**.

The Notifications chip then says **Linked to:** with that name. **Telegram only counts as set up once a chat is linked:** a token on its own sends nothing, and **Send test** skips Telegram until you have linked. **Unlink** forgets the chat; press **Link...** again to link a different one. A group chat works too: add the bot to the group, then send `/start` in the group.

If linking says the bot already has a **webhook** set, another program is receiving that bot's messages. NightFront offers **Remove webhook**; only use it if the bot is meant for NightFront alone. If you paste the token of a different bot, the old chat link is cleared, because a chat belongs to the bot it was linked through. If Telegram later says the bot was blocked or removed from the chat, the page tells you to press **Link...** again, and Telegram messages stop until you do, even after NightFront restarts (Pushover, if set up, carries on).

Setting up Pushover

1. Create an account at pushover.net. Your **user key** is shown on the dashboard.
2. Install the Pushover app on your phone. **The receiving app is a one-time purchase of \$4.99 per platform (iPhone, Android, desktop) after a 30-day free trial.** Sending is free: Pushover's free tier allows 10,000 messages a month per application, far more than processing needs.
3. On pushover.net, choose **Create an Application/API Token**, name it (for example, NightFront) and copy its **API token**.
4. In NightFront, paste the token into **Application token** and your key into **User key**, then press **Validate**. NightFront asks Pushover whether the key is good and reports the answer under the button.

Testing it

Send test sends one test message to every channel that is set up, so you can see it arrive on your phone. It works whether or not the master switch is on and whatever steps are ticked; it only needs at least one channel to be configured. Under each channel the page shows the result of the last message sent, and a message that could not be delivered is reported on the Processing page (with **Dismiss**) instead of being lost. NightFront retries a message at most once: when it could not connect at all, when the service reports a server error, or when Telegram asks it to wait a few seconds. It never retries after a message went out and no answer came back in time, since the service may already have accepted it, so a slow network does not give you duplicates.

Your tokens and keys never appear in any message, error text or log NightFront writes.

Limits worth knowing

- **Only NightFront running on your machine sends.** If you send a job to a processing server and then close NightFront or let the computer sleep, nobody is notified while the job finishes on the server. The next time NightFront runs, it collects the finished job and sends **Remote job finished**. If you closed the tab while a remote job was running and the job's result was still brought back in, that also counts as a real ending and sends a message — even for a project in **Calibrate all** or **Stack all**, or the calibration a **Stack** starts with: with the batch or the stack cut off, that job sends its own message (for example **Calibration finished**) instead. A job NightFront merely lost contact with sends nothing until its real result is known.
- **Your machine needs internet access to send.** A message that cannot be delivered is shown on the Processing page and dropped; it is not queued for later.
- **Quitting right after a step ends can cut its message off.** When you close NightFront it waits up to five seconds for messages still being sent, then exits; a message that has not gone by then is lost, and with the app closed nothing is left to report it.
- **Going back to an older NightFront loses the settings.** They live in your user preferences file. An older version does not know about them, and the next time it saves its own preferences it drops them. After updating again, set them up again.
- **The tokens are stored as plain text** in your preferences file (`preferences.json` in the `.config/nightfront` folder of your user profile), the same as processing-server peer tokens. Anyone who can read your user folder can read them, so use a bot dedicated to NightFront. If a token leaks, revoke it (BotFather's `/revoke`, or delete the Pushover application) and paste the new one.

See [Processing](#) for the steps these messages report on, and [Remote processing](#) for jobs that run on another machine.

Running with NINA

The desktop app decides what a night should optimize for. The **NightFront NINA plugin** is the half that runs on your imaging computer, inside NINA: it gets a plan into your NINA sequence at dusk, keeps it answering to the real sky through the night, and counts what it shoots so progress carries across unattended nights.

This page is an overview. The plugin has its own guide with every instruction, trigger and setting in detail — see [The plugin guide](#) at the foot of this page.

What the plugin does in NINA

The plugin adds these items to NINA's Advanced Sequencer:

Item	Kind	What it does
Nightly Update	Instruction	Gets tonight's plan into the NightFront Container. If a plan file for tonight is in the plugin's data folder (one you exported from the app), it imports it. If it already generated a plan earlier tonight (a sequence restarted after a safety stop, say), it re-imports that one. Otherwise it generates one on the spot from your NightFront config, the forecast and live readings from your weather device, within the solve time you set on the instruction (If generating, plan for up to ... , 180 seconds by default).
NightFront Container	Container	Holds the night. Nightly Update and Replan fill it with ordinary NINA instructions — slew, center and rotate, filter changes, exposures, dithers — which NINA runs like any hand-built sequence. It shows a live per-target summary as the night goes.
Replan	Instruction	Re-solves the rest of the night : records what has been captured so far, takes a live weather reading, runs the planner again and swaps the new plan into the container. Can name an Exposure variant (for example "High Wind") to shoot shorter subs.
Seeing Trigger	Trigger	Watches your weather device's star-FWHM sensor and runs what you drop into it — typically a Replan — when the reading crosses your threshold.
Wind Speed Trigger	Trigger	Two drop areas, one for the wind rising past your threshold and one for it falling back. The usual wiring is a Replan with a "High Wind" exposure variant above, and a plain Replan below.
Before Target / After Target	Triggers	Fire once around each NightFront target, so you can hang notifications (a Ground Station message, say) off them.
While Work Remains	Loop condition	For the outermost loop of an unattended sequence: keeps looping while any enabled Active target still needs integration time, and stops once they are all done.
NightFront Sky Flats, NightFront Trained Flats, NightFront Rotate to Next Angle, NightFront While Flats Remain, NightFront While Same Rotation	Calibration	Shoot flats matching the rotator angles, gains, offsets and readout modes your lights were actually taken at.

Refocusing is left to your own sequence-level autofocus triggers; the plugin adds none.

When the plan runs out with darkness left, the plugin refills the night on its own rather than leaving the mount idle; the plugin guide's "When the plan finishes early" section describes how, and the switch on the plugin's Options tab turns it off.

Heads up: Replan does not re-solve a night that was not planned by the optimizer. If tonight's plan came from the app's Editor, or from the Greedy, Proportional or Dispatch planner, or from the plugin's own Greedy or Proportional mode, Replan leaves it alone — re-solving would replace a plan you chose with an optimizer plan, unattended. Tick **Replan hand-built plans** on the Replan instruction if you want it to re-solve those anyway.

A recommended NINA sequence

A typical unattended sequence, built from NINA's usual safety template, looks like this. It is the shape of the example night the plugin's installer puts on your Desktop. The names in quotes are containers you create yourself, and the conditions named are NINA's own or the plugin's; yours may differ.

```
Sequence
├── "Every night" (loop)                                condition: While Work Remains
│   ├── "Start of night"
│   │   ├── Wait for Time: sunset
│   │   ├── ...your own startup: connect, cool the camera, open the roof...
│   │   ├── Wait for Time: nautical dusk
│   │   ├── ...your own first focus: Unpark Scope, slew to a star field, autofocus...
│   │   └── Nightly Update
│   ├── "Imaging until dawn" (loop)                    condition: Loop until nautical dawn
│   │   ├── "Imaging" (loop)                          condition: Loop while safe
│   │   │   ├── triggers: Seeing Trigger → Replan
│   │   │   │   Wind Speed Trigger → above: Replan (Exposure variant "High Wind")
│   │   │   │   below: Replan
│   │   │   └── Before Target / After Target → notifications
│   │   │       your own autofocus triggers
│   │   └── NightFront Container
│   └── "Safety recovery"
│       ├── ...your own weather-stop steps: Park Scope, close the roof...
│       ├── "Wait while unsafe" (loop) condition: Loop while unsafe
│       ├── ...your own recovery steps: reopen the roof, Unpark Scope, cool the camera...
│       └── Replan
├── "Sky flats" (loop)                                conditions: While Flats Remain, Loop while safe,
│   └── Loop until sunrise +30 min
├── "End of night"
│   ├── Park Scope
│   ├── ...your own shutdown: close the roof, warm the camera...
│   └── NightFront Save Night's Log (last)
└── Wait for Time: sunrise, offset +5 minutes
    └── End area: anything for when the whole sequence stops
```

Rules that matter:

- **Nightly Update must sit in a container that runs once, near the start of the night, and the NightFront Container must come after it in the same branch.** The container does not have to be a direct sibling — it can be nested inside a repeating loop, as above.
- **Imaging and its safety recovery sit together inside the loop that runs until dawn,** with the recovery after imaging. As separate branches of the night, the first weather stop would fall through to the end of the night, and flats, park and Save Night's Log would run in the middle of it.
- **Every Park Scope needs an Unpark Scope after it, before the next imaging.** A parked mount refuses every slew: unpark in the safety recovery after the wait, and in the next night's first focus.
- **Replan can go anywhere that can reach the container** — typically inside the weather triggers, and in the safety recovery that runs when the roof reopens, so a cloud-out does not simply resume a plan that no longer fits the night.
- **Before Target and After Target must be placed on the NightFront Container itself or a container that encloses it** (such as "Imaging"), or they never see its targets begin. A sibling branch such as the safety recovery does not work.

- **Each night opens with a wait for sunset and closes with a wait for a few minutes past sunrise**, so each pass of "Every night" starts in the evening. The closing wait needs its offset.
- **Dawn flats need a time limit.** After sunrise a sky flat fails and stays needed, so a flats loop with no Loop until condition would retry it all day.
- **While Work Remains belongs on the outermost loop** — the one that opens the roof each evening. When it cannot tell (an unreadable config, say), it keeps looping rather than ending the night.

The plugin's installer puts a complete, annotated version of this on your Desktop, in the **NightFront NINA Templates** folder: an example night and five templates, built only from NINA's own items and NightFront's. The plugin guide describes them.

Plugin settings that shape a night

These live on the plugin's own options page in NINA, on its **Setup** page.

Setting	What it does
NightFront Data Folder	Where plan files are imported from and where the plugin keeps its own records. Leave it clear to use the folder that holds your NightFront config, and export from the app into that folder too.
NightFront Config (for online planning)	The NightFront config the plugin plans from. See Which config the plugin reads .
Planning Mode	How Nightly Update builds a night when it has to generate one: Optimizer (GA) (the default), Greedy or Proportional . The two deterministic planners follow your target priority order and finish instantly; the solve time and Plan Selection have no effect under them.
Plan Selection	Which point on the optimizer's Pareto front a night lands on: Max Quality, Balanced (the knee of the utilization/quality curve, and the default) or Max Utilization . It applies to both Nightly Update and Replan, so a mid-night replan cannot quietly switch philosophy.
Custom Horizon (from NINA)	Read-only: the horizon file set in NINA's own Options → Astrometry. It is used only when your NightFront config states no horizon of its own.

Note: The plugin's Planning Mode offers the optimizer, Greedy and Proportional. The app's Dispatch planner is not a plugin mode; to run a Dispatch night, generate it in the app and export it.

The same options page has a **Targets** page for checking progress and editing requested subframe counts on the imaging computer, a **Check tonight** feasibility check, and a way to create targets from NINA's Framing Assistant. Those are covered in the plugin guide.

Which config the plugin reads

Point **NightFront Config (for online planning)** at the `.json` you save from the desktop app — the same file you open in the app.

Heads up: Do not point it at `session-config.json`. That file is a snapshot the app's **Export Sequence...** writes beside a plan, and it is frozen at the moment of export: its accepted counts never move again. If the setting is left empty the plugin falls back to that snapshot, and a stale copy re-schedules data you already

have while reporting success. For the same reason, Nightly Update shows an amber advisory when the config it plans from was last saved 7 or more days ago. (The setting's **Clear** button returns it to empty, and so to that fallback.)

Because the plugin reads the config you save, **a change you make in the app reaches tonight's planning as soon as you save it** and the file reaches the imaging computer.

Sharing files between the desktop and the imaging computer

The desktop app and NINA usually run on different computers: the app on the machine you work at, NINA on the imaging computer at the telescope. **They must see the same NightFront files.** Put them in a folder that both computers share and keep in sync: a cloud drive such as Google Drive, OneDrive or Proton Drive, a peer-to-peer sync tool such as Syncthing, or a network share both can reach.

What has to live in that shared folder:

- **Your NightFront config** (the `.json` you save from the app) and the files NightFront keeps beside it. The plugin plans from this file, and its **Targets** page and Framing Assistant changes write back to it.
- **The plugin's NightFront Data Folder.** Plans you export from the app land here for Nightly Update to import, and the plugin keeps its records of the night here. Leave it clear and it is your config's folder. The plugin saves the night logs that the app's **Analysis** page reads beside the config either way. If you set the data folder to another folder, an amber warning on the plugin's Setup page and on Nightly Update says so, and the plugin asks first before any change that would leave its ledger of collected frames behind.

On each computer, point at the synced folder's local path: on the desktop, save the config there and choose it in **Export Sequence...**; on the imaging computer, set the plugin's **NightFront Config (for online planning)** to the same file as it appears on that machine, and leave its **NightFront Data Folder** clear.

Heads up: A sync tool only helps once it has finished syncing. Save in the app well before dusk, and make sure the imaging computer is online and its sync client running, so tonight's plan starts from your latest changes rather than yesterday's copy.

Heads up: Both sides can change the config: the app when you save, the plugin when you save on its **Targets** page or create a target from the Framing Assistant. Whichever saves last wins, and some sync tools keep the other version as a separate conflict copy that nothing reads. Avoid editing the same config in both places at once; after changing it from NINA, press **Reload from disk...** on the app's Targets page before you edit and save there.

If the app and NINA run on the same computer, no sync is needed; just use the same folder for both.

The frames themselves travel the other way, from the imaging computer back to the desktop for [Processing](#); the same kind of shared folder works for that.

Forecasts on the imaging computer

The planner inside the plugin uses the same forecast source as your config — 7Timer! by default, or Astrospheric with an API key — and caches it on disk the same way the app does, per source: a forecast is reused for 3 hours from 7Timer! and 6 hours from Astrospheric, so a night of repeated replans does not re-fetch it. (If the app and NINA run on the same computer under the same Windows account, they share that cache.)

When a fresh forecast cannot be fetched, the planner falls back to a cached one up to 48 hours old and says the night rests on an out-of-date forecast. With nothing usable, it reports that no forecast is available and leaves the plan already in the container in place rather than inventing weather.

Live readings from your weather device override the forecast only for the hours around the moment they were taken.

The plugin plans as many nights as the **Horizon** setting saved in your config (Schedule › Nightly › Setup; one night if the config has never stated one), but it only ever loads tonight's plan into the container. Later nights never change tonight's schedule.

The plugin guide

Everything above is covered in more depth — each instruction's fields, the Options tab's Targets page and **Check tonight**, creating targets from the Framing Assistant, calibration flats, and the collected-frames tally — in the plugin's own guide. In NINA, open NightFront's options page (**Options** → **Plugins** → **NightFront**) and press **Plugin Guide** at the top of its **Setup** page.

Command-line summary

The planner the plugin runs is a command-line program, `nightfront-cli.exe`, installed inside the plugin's folder. The plugin's planning, its feasibility check and its edits to your config all go through it, and you can run it by hand. This is a summary for power users; running `nightfront-cli.exe` with no arguments prints the full usage.

Flags always take the form `--name=value`. An unrecognized flag is an error.

```

nightfront-cli <config.json>

nightfront-cli plan [flags] <config.json> <live-conditions.json|none>
                    <output-plan.json> [selection.json]

nightfront-cli replan [flags] <config.json> <progress-snapshot.json>
                    <live-conditions.json|none> <output-plan.json> [selection.json]

nightfront-cli feasibility [flags] <config.json> <output-report.json>

nightfront-cli set-planned --target=<name> --filter=<name> [--exposure=<seconds>]
                    (--planned=<subframes> | --planned-hours=<hours>) <config.json>

nightfront-cli add-targets --template=<name>
                    (--target-name=<name> | --mosaic-name=<name>)
                    --panel="<raDeg>|<decDeg>|<rotationDeg>" [--panel=...]
                    [--min-imaging-minutes=<n>] <config.json>

```

- **<config.json> alone** — solves the config with the optimizer and prints the chosen schedule. Writes no plan file.
- **plan** — plans tonight from scratch (from now until dawn) and writes a NINA plan file. What Nightly Update runs when it generates.
- **replan** — re-solves the remainder of a night already under way, given what has been captured. What Replan runs.
- **feasibility** — writes, per Active target, how many hours are imageable across the whole of tonight or why none are. No planning run. What **Check tonight** runs.
- **set-planned** — changes one filter row's requested amount (in subframes or hours) and saves the config the same way the app's own Save does. What the plugin's Targets page **Save** runs.
- **add-targets** — adds one target, or a mosaic with one **--panel** per panel, using an exposure template from the config. What the Framing Assistant **Create** runs.

Key flags for **plan** and **replan**:

Flag	Meaning
<code>--time-budget=<seconds></code>	Wall-clock solve budget. <code>plan</code> defaults to 180 seconds; <code>replan</code> uses its effort level unless you give one.
<code>--effort=<level></code>	<code>replan</code> only (refused by <code>plan</code>): a fixed solve size — <code>fast</code> (the default), <code>balanced</code> or <code>thorough</code> .
<code>--strategy=<planner></code>	<code>plan</code> only (refused by <code>replan</code> , which always uses the optimizer): <code>ga</code> (the default), <code>greedy</code> , <code>proportional</code> or <code>dispatch</code> . An unrecognized value falls back to the optimizer with a warning.
<code>--plan-selection=<point></code>	Which Pareto-front point to export: <code>balanced</code> (the default), <code>max-quality</code> or <code>max-utilization</code> .
<code>--nights=<n></code>	How many nights, 1 to 7, to plan across; only the first is exported. Without it, the config's saved Horizon decides (one night if it states none).
<code>--night-date=YYYY-MM-DD</code>	Plan a specific observing night instead of the one the clock says.
<code>--collected=<file></code>	The plugin's collected-frames tally, subtracted from what each target still needs.
<code>--exposure-variant=<name></code>	Shoot at a named exposure variant from the config. An unknown name plans at the default exposures.
<code>--horizon=<file.hrz></code>	A NINA custom-horizon file, used only when the config states no horizon.
<code>--schedule-report=<file></code>	Also write a readable report of the schedule, for diagnosis.

`feasibility` accepts `--night-date`, `--collected`, `--nights` and `--horizon`, and diagnoses one night unless told otherwise.

Note: `plan` and `replan` exit successfully even when there is nothing to plan, no darkness left, or no forecast; the outcome is written beside the output as `<output-plan.json>.result.json`, so an ordinary quiet night never aborts a NINA sequence. `feasibility` likewise reports those three outcomes inside its report file. All five exit with an error for a real mistake — a bad flag, a missing or malformed config — and `set-planned` and `add-targets` also exit with an error whenever the change could not be made.

Files and data

NightFront keeps your work in three kinds of place: **config files** you choose the location of, small **sidcar files** it writes beside those configs and beside exported plans, and a **per-user folder** of settings and caches. This page says what is where, what is safe to delete, and what survives an upgrade.

Your config files

Each imaging system is one `.json` config, saved wherever you choose with **Save As...** (see [Getting started](#)). It holds everything about that rig: site, forecast source and API key, filters, moon profiles, exposure templates and variants, horizon, targets and their accepted counts, and every planning setting.

The config is the file to back up. It is also the file the NINA plugin should plan from, so when NINA runs on another computer the config (with the files beside it) and the plugin's data folder belong in a folder both computers share and sync — see [Sharing files between the desktop and the imaging computer](#).

Heads up: The config holds your Astrospheric API key, if you entered one. Keep that in mind before sharing a config with someone else.

Sidcar files beside a config

When a config has been saved, NightFront keeps these beside it, named after the config. For a config called `E160.json`:

File	What it holds	If you delete it
<code>E160.plan</code> - <code>variants.json</code>	Every plan in the Editor — hand-built and generated — and which one was selected. Updated as you work.	Those plans are gone. The Pareto front is not stored here and is not affected.
<code>E160.seasonal-plan.json</code>	The last roadmap run on the Schedule › Seasonal › Plan page.	The roadmap is gone until you run it again.
<code>E160.overhead-samples.jsonl</code>	The running record of overheads read from your NINA logs (slews, flips, dithers, autofocus runs and what was collected each night), plus the dates of any eras you started and any discard-before date. Only exists once the NINA plugin has saved a night's log into the nina-logs folder beside the config (see Analysis).	Safe to delete. The record starts over from the nights still saved in nina-logs ; nights no longer there cannot be rebuilt.
<code>E160.overhead-plans</code> (a folder)	Copies of the NightFront plan files found for those nights, kept before the NINA plugin removes the originals.	Safe to delete. Nights scored against those plans then read <i>Actual only</i> ; plans the plugin has already removed cannot be rebuilt.

Older builds named the seasonal file `E160.longrange-plan.json`. NightFront still reads that name if the new one does not exist, and writes the new name the next time you run a roadmap or save the config; the old file is left behind and can then be deleted.

Sidecars travel with the config: copy or move the folder and the plans come along. The plan-variants file holds no copy of your targets — those plans are rebuilt against the config every time it opens. The seasonal plan, by contrast, is a saved result, including a snapshot of the targets it was run against.

Files written by an export

Export Sequence... on the Schedule › Nightly › Plan page asks for a folder — normally the NINA plugin's data folder — and writes:

File	Written	What it is
<code>TargetsForTonight_<date>.json</code>	One per night in the plan	The NINA plan for that night, which the plugin's Nightly Update imports. The date is the local date the night starts.
<code>TargetsForTonight_<date>.authorship.json</code>	One per night	Who made the plan: the optimizer, a generated planner, or a hand-built plan, plus the plan's name. The plugin reads it to decide whether Replan may resolve the night. A generated plan you have edited counts as hand-built.
<code>selection.json</code>	Once per export, only when the exported plan is a point on the Pareto front	The utilization/quality balance you picked, so a later Replan can stay near it.
<code>session-config.json</code>	Once per export	A frozen snapshot of the config at the moment of export.

Heads up: `session-config.json` is a snapshot, not your config. Its accepted counts never update. Point the plugin at your real config file instead; see [Which config the plugin reads](#).

Exporting again to the same folder overwrites these files. Each night's authorship marker is always rewritten (or removed) along with its plan, so an old marker never describes a newer plan. The one exception is `selection.json`: an export of a plan that is not on the Pareto front writes none, so an earlier export's `selection.json` stays in the folder.

The plugin keeps its own records in its data folder too — the frames it has collected, calibration requirements for flats, archived replans — and saves the night logs that **Analysis** reads into a `nina-logs` folder beside the config it plans from. Those are described in the plugin guide.

Heads up: Keep your config, your export folder and the plugin's data folder as **one folder**. The plugin's data folder does that by itself when you leave it clear (it then uses the config's folder). The plugin keeps the collected-frames ledger and the flats queue in its data folder, and **Queue flats...** on the Processing page writes to the folder you last exported to, so a data folder set somewhere else puts the ledger and the queued flats apart from the config without any error.

The NightFront settings folder

NightFront keeps per-user settings and caches in a `.config/nightfront` folder in your home directory:

- **Windows:** `C:\Users\<you>\.config\nightfront\`
- **Linux and macOS:** `~/.config/nightfront/`

On the imaging computer, the planner inside the NINA plugin uses the same folder under the Windows account NINA runs as — which is how its forecast cache is shared across a night of replans.

What is in the settings folder

Item	What it holds	Safe to delete?
<code>preferences.json</code>	Which configs were open, recent files, last-used folders, window size and position, the PixInsight and Siril paths, the calibration engine, the Projects root and working folder (all set in App settings) (and every working folder used before), each imaging system's processing settings (flat integration, stacking, download folder), this machine's own remote-processing server settings (bind address, port, token, serve-at-launch — keyed to this machine's hostname) and every peer you've added	No — this is settings, not cache. Deleting it resets all of the above, including your processing server's token (every peer would need it again) and your peer list.
<code>remote-jobs.json</code>	Remote processing jobs this machine has sent to a peer and not yet collected	No , while any are pending — deleting it lifts every hold those jobs placed and leaves the peer holding a job this machine no longer remembers, reachable again only through the peer's own retained record until it's collected or abandoned there directly.
<code>preferences.json.corrupt- <timestamp></code>	A copy of a <code>preferences.json</code> NightFront could not read, set aside before starting from defaults	Yes, once you have recovered anything you need from it
<code>sites.json</code>	Your Sites Library	No — deleting it empties the library (sites already in configs are unaffected)
<code>forecast-cache.json</code>	Recent forecasts, per site and per forecast source	Yes. The next load fetches fresh — which costs Astrospheric credits if you use it — and the planner loses its fallback for a night when the network is down.
<code>simbad_cache.json</code>	Results of SIMBAD object lookups	Yes. Lookups query SIMBAD again.
<code>climate-cache.json</code>	NASA POWER climate profiles, per site	Yes. You may need to fetch a site's climate profile again.
<code>catalog-cache.json</code>	The Discover Targets catalog (several megabytes)	Yes, but Discover Targets then has to download all its catalogs again before it can search.
<code>nightfront-err.log</code> , <code>nightfront-</code>	The crash and error log; rolls over to <code>.log.1</code> at 4 MB	Yes — but if something is wrong, send it first. See Troubleshooting .

Item	What it holds	Safe to delete?
<code>err.log.1</code>		
<code>header-cache/</code>	Per rig folder, the FITS/XISF headers read during a Processing scan	Yes. The next scan re-reads every header, which is slow over a network share.
<code>measurement-cache/</code>	Per rig folder, frame measurements from the Processing review pass	Yes, but frames are measured again, which takes time.
<code>blink-drafts/</code>	Per rig folder, blink verdicts you have made but not yet committed	No , unless you are happy to lose those verdicts.
<code>runs/</code>	The default Working folder (intermediate frames) for Processing runs: calibration and stacking intermediates, scripts and run records. On a machine set up as a remote-processing server, a peer's job runs under <code>remote jobs\<job id>\</code> inside it, cleaned up as soon as the client collects the result; on a machine sending work away, the collected result briefly stages under <code>remote ingest\<job id>\</code> on its way into the ordinary run folder.	Not freely. NightFront reads the run records here to know which frames are already calibrated; deleting them makes projects queue for calibration again. See Processing .
<code>last_session.txt</code>	Left over from very old builds	Yes

Note: If you set a different **Working folder (intermediate frames)** in App settings > Paths, runs go there instead of `runs/`, and the same caution applies to that folder.

Data inside your Projects tree

The Processing tab also writes a few small files into your Projects tree — most importantly `nightfront-processing.json` in each rig folder, which holds your processing work for that rig (which configs it serves, project links, annotations and overrides). Treat it like a config: it is your work, not a cache. It also writes three small hidden marker files. Don't delete them:

File	Where	What it is	If you delete it
<code>.nightfront-rig-id</code>	In each rig folder	The rig folder's permanent id. NightFront's records of what it has calibrated there, its blink verdicts and its cached headers are filed under this id rather than the folder's name, so the folder can be renamed. Written the first time Processing scans the folder.	The next scan gives the folder a new id, and the records filed under the old one are no longer found: projects queue for calibration again and blink verdicts are lost.
<code>.nightfront-project.json</code>	In each project folder	The project's permanent id, so its reviewed and stacked frames, stacking settings, target link, object-name rulings and unsent blink verdicts survive a rename, in NightFront or in File Explorer. Written the first time Processing scans the folder, or when NightFront creates it.	The project reads as never reviewed and never stacked until it is scanned again, and a renamed folder is not recognized.
<code>.nightfront-remote-job</code>	In a project folder (or the rig folder, for a masters build)	Present only while a job sent to a processing server is working there and has not been collected or abandoned. Other computers that reach the same Projects folder see it and refuse to rename, merge, complete, restore or move that project meanwhile.	Lifts the hold. Delete it by hand only if the computer that sent the job is gone for good; otherwise the job's result may land in a folder you have already moved.

Copying a project folder in File Explorer copies its marker too, so the two copies claim one id. NightFront reports it and asks rather than guess which is the copy; delete `.nightfront-project.json` from the copy and **Rescan** to give it its own id.

Upgrades and uninstalling

- **Your data lives outside the install folder.** NightFront keeps its settings and caches in the `.config/nightfront` folder in your home directory, and your configs, their sidecars, exported plans and your Projects tree wherever you put them — none of it inside the folder the installer manages. Installing a newer version replaces the program itself.
- **To remove NightFront's settings and caches entirely,** delete the `.config/nightfront` folder yourself after uninstalling.
- **Moving to another computer:** copy your config files (with their sidecars) and, if you want your Sites Library and app settings too, `sites.json` and `preferences.json`. Paths stored in `preferences.json` (PixInsight, Siril, the Projects root) describe the old machine and may need setting again.

Troubleshooting

Start with the error log if the app itself misbehaved; the rest of this page is organized by symptom.

NightFront froze or closed unexpectedly

The installed app has no console, so errors go to a log file instead:

- **Windows:** `C:\Users\<you>\.config\nightfront\nightfront-err.log`
- **Linux and macOS:** `~/.config/nightfront/nightfront-err.log`

Each launch adds a `=== NightFront <version> started ... ===` line, followed by any errors from that session. When the file passes 4 MB it is renamed to `nightfront-err.log.1` at the next launch and a new one begins.

If you report a freeze, a crash or a window that vanished, send `nightfront-err.log` (and `nightfront-err.log.1` if it exists) first, with roughly when it happened. It is the only record of what went wrong.

Note: This log covers the desktop app. Problems with planning inside NINA are logged by NINA and the plugin; see [Problems inside NINA](#).

Windows install or upgrade fails with error 1603

NightFront installs for all users of the machine, so **both installing and upgrading need administrator rights**. Double-clicking the `.msi` normally prompts for them. If you run `msiexec` from a command prompt or a script, run it from an **elevated** (Run as administrator) prompt — an unelevated one fails with error 1603, and on an upgrade the log shows error 1730 underneath it.

You do not need to uninstall the old version first; the new `.msi` replaces it.

The AppImage will not start

On some modern Linux distributions an AppImage needs the `libfuse2` library, which is not always installed. Either install it (on Debian and Ubuntu the package is usually `libfuse2` or `libfuse2t64`) or run the AppImage with:

```
./NightFront-<version>-x86_64.AppImage --appimage-extract-and-run
```

Also check the file is executable (`chmod +x`). If neither helps, the `.deb` is the alternative.

macOS says the app cannot be opened

The macOS build is **not signed**, so Gatekeeper refuses a plain double-click. Right-click (or Control-click) **NightFront** in Applications, choose **Open**, and confirm. You only need to do this once. If macOS still refuses, run this in Terminal and try again:

```
xattr -dr com.apple.quarantine /Applications/NightFront.app
```

The `.dmg` is built for Apple Silicon and **will not run on an Intel Mac**.

The forecast will not load

On the [Site](#) page the forecast banner names the source in use — **7Timer! ASTRO** or **Astrospheric** — and when a load fails, the reason appears in red just below it, with **No forecast data. Click Refresh.** where the chart would be.

- **Check the connection, then press Refresh.** The desktop app does not fall back to an older forecast when a fetch fails; it shows the error.
- **7Timer! (the free default)** is a single free service. It is occasionally slow or down; try again later. A message that the upstream model "appears stuck" means 7Timer! is serving a forecast more than a day old, which NightFront refuses rather than plan against.
- **Astrospheric** needs a valid API key in **Astrospheric API Key**. If the source is set to Astrospheric but no key is entered, NightFront quietly uses 7Timer! instead — the banner above the chart shows which source is really in use. Astrospheric charges credits per call from your monthly allowance; the **Refresh** button shows the cost of one call (for example **Refresh (65 credits)**). NightFront does not display your remaining credits — check your Astrospheric account if calls start failing.
- **Avoid pressing Refresh repeatedly.** An ordinary load reuses a cached forecast (up to 3 hours old for 7Timer!, 6 hours for Astrospheric). **Refresh** always fetches, and on Astrospheric always costs credits.

On the imaging computer the planner is more forgiving: if a fetch fails it plans against a cached forecast up to 48 hours old and says so, and with nothing usable it reports that no forecast is available and leaves the existing plan in place. See [Forecasts on the imaging computer](#).

A processing peer is unreachable

Adding a peer, or dispatching a job to one, reports it as unreachable. Nothing outside that machine can tell these three apart:

1. **It isn't serving.** On that machine, open **App settings**, choose the **Server** chip and confirm **Serve processing jobs** is on.
2. **The address, port or token don't match.** Compare them against exactly what that machine's Server chip shows — a regenerated token invalidates every peer still configured with the old one.
3. **A firewall is blocking it.** The first time a machine turns on Serve, its OS firewall typically asks whether to allow NightFront; if that was denied (or never asked, on some Linux firewalls), no peer can reach it. Check that machine's firewall rules for NightFront.

See [Remote processing](#) for setting up a server and a peer.

A target gets no time

Work down this list.

1. **Is it eligible tonight?** Only **enabled Active** targets are planned. A target that is Planned, disabled, completed, or has no filters — or whose every filter already has as many Accepted subframes as Planned — is skipped. See [Targets](#).
2. **Run Tonight's Feasibility.** On Schedule › Nightly › **Setup**, press **Check** under **Tonight's Feasibility**. It needs no planning run, and sorts problems into:
 - **No feasible time** — the target (or a filter) is never usable on the forecast nights: never high enough, too close to the moon, or high enough and clear of the moon at different times but never both at once.
 - **Partly blocked** — some filters, or some nights, are blocked.
 - **Available, but never long enough** — the target is usable, but never for an unbroken stretch as long as its **Min imaging time**. This is the easy one to miss, and it is common on mosaics, where each panel's window is short. Lower the minimum imaging time on that target.
3. **Check the sky limits.** **Min Altitude** and the custom horizon (both on Imaging System › **Horizon**), and each filter's moon avoidance profile, are hard limits. A horizon or a strict narrowband profile around full moon can shut a target out entirely. See [Imaging System](#).
4. **If it is feasible and still gets nothing**, the planner simply found better uses of the night. Try another point on the Pareto chart (utilization-heavy plans favor fewer targets), or press ► **Greedy** with the target high in your priority order, and compare. See [Planning a night](#).

Generated plans disappeared

If a Greedy, Proportional or Dispatch plan vanished from the chart and the Editor, look for this note on the Schedule › Nightly › **Plan** page:

Removed 1 generated plan — the setup changed since it was built.

A generated plan is rebuilt in seconds, so NightFront removes one as soon as its inputs change — your targets, priority order, planning settings, or a newer forecast — rather than leave a plan on the chart that no longer matches your setup. Press the planner's button again to regenerate it.

- **Plans you have edited by hand are never removed this way**, and neither are plans you built yourself in the Editor.
- **The GA's Pareto front is never removed automatically**, because re-running the optimizer takes minutes. If you change the minimum altitude or the horizon after a run, the Plan page says the front describes the previous horizon and asks you to re-run ► **GA**.

Recent files and settings were reset

If your recent files, open tabs, PixInsight path, or processing settings suddenly reverted to defaults, NightFront may have found `preferences.json` unreadable. It does not overwrite such a file: it moves it aside as `preferences.json.corrupt-<timestamp>` in the same folder, starts from defaults, and writes a line explaining why to `nightfront-err.log`. Your old settings are still in that file if you need to recover a path from it. See [Files and data](#).

Problems inside NINA

For problems inside NINA — Nightly Update not importing or generating a plan, Replan doing nothing, triggers that never fire, the Targets page or **Check tonight** — see the plugin's own guide, opened with the **Plugin Guide** button on NightFront's options page in NINA. Two common causes worth checking first:

- The plugin is reading the frozen `session-config.json` snapshot instead of your saved config. See [Which config the plugin reads](#).
- Replan deliberately does nothing when tonight's plan was hand-built or generated rather than optimized, unless **Replan hand-built plans** is ticked. See [Running with NINA](#).

Still stuck

Ask on NightFront's Discord server: <https://discord.gg/65ZYummSwb>, or the **Community** button at the foot of the app's left-hand rail.

Bring the two things that make a problem answerable rather than guessable: `nightfront-err.log` for anything the app itself did, and NINA's own log from the night in question for anything the plugin did. The **Report** chip of the [Analysis](#) page turns NINA's logs into a scrubbed support report that you review in full, then save or copy; NightFront never sends it for you. Say which versions of the app and the plugin you are running — the app's is on the title bar, the plugin's is in NINA's Plugins list — since the two ship separately and may differ.

Glossary

Short definitions of the terms NightFront uses, with the page that covers each one.

Accepted — The subframes you have reviewed and kept for a target, per filter — the **Accepted** column on the Targets page. NightFront plans only what is still needed: **Planned** (called Requested in the plugin) minus Accepted (and, on the imaging computer, minus what the plugin has collected since your last review). Accepted counts come from accepting frames on the Processing tab, or entering them by hand. Moving or reframing a target resets them to zero. See [Targets](#) and [Processing](#).

Active / Planned — A target's category. **Active** targets are in the rotation now and are the only ones tonight's planning ever considers. **Planned** targets are on the roadmap, waiting; Schedule › Seasonal › Plan weighs both. See [Getting started](#).

Airmass — How much atmosphere the light passes through, relative to straight overhead (1.0 at the zenith, larger toward the horizon). Lower is better, and it is the main ingredient of the **Quality** objective. Shown per slot on the Plan page. See [Planning a night](#).

Analysis — The rail page that reads NINA's own log files: last night's report, overhead and autofocus recommendations and a scrubbed support report. See [Analysis](#).

Autofocus, dither and meridian flip overhead — Time the plan spends not exposing: slewing and centering on a target, meridian flips, dithers at each filter's cadence, and autofocus runs. NightFront schedules these as real time, so they count against **Utilization**; they appear as gray bars in the Gantt. Autofocus time is planned for but not exported — refocusing is left to your own NINA triggers. **Max Slew/Center/Rotate is an upper bound, not a flat rate**: a nearby move, or one needing little rotation, is charged half, both together a quarter, and staying on the same target nothing at all. See [Imaging System](#).

Blink — The Processing step where you step through a night's new lights frame by frame and accept or reject each one before they join the project. See [Processing](#).

Completion — Two related things. *Completion Priority* is a slider on Schedule › Nightly › Setup that makes the optimizer favor targets that are nearly finished over spreading time thinly. *Completing a project*, on the Processing tab, retires it: it leaves the Active and Planned lists and appears under **Completed** on the Targets page, where **Plan again** brings it back. See [Planning a night](#) and [Processing](#).

Dispatch — A rule-scoring planner: at each decision point it scores every candidate by weighted rules (priority order, percent complete, setting soonest, stay on target and more) and takes the highest. Its rule weights are under **Dispatch planner rules** on Schedule › Nightly › Setup. See [Planning a night](#).

Dwell — How long the optimizer stays on a target once it has started one. NightFront's optimizer can hold a target beyond its minimum imaging time, which favors fewer, longer visits and fewer slews. There is no setting for it. See [Planning a night](#).

Editor — The chip on Schedule › Nightly › Plan where a plan is a set of blocks on a timeline that you can build, move, resize and restrict to particular filters. Generated plans open there too, so any plan can be adjusted by hand. See [Planning a night](#).

Era — A stretch of nights over which an overhead measurement still describes how your rig runs. A change that could move it (an Apply, a hand edit, a new NINA setting) starts a new one, and older nights stop counting toward that suggestion. See [Analysis](#).

Exposure template — A reusable exposure plan — filters, sub lengths, counts, gain, offset, dither cadence, moon profiles — defined once on Imaging System > **Exposures** and loaded into targets. See [Imaging System](#).

Exposure variant — A named set of per-filter sub-length overrides (for example "High Wind": Ha at 300 s instead of 600 s), defined on Imaging System > **Variants** and chosen at run time, typically by a Replan fired from the plugin's Wind Speed Trigger. The same sky time is kept, divided into more, shorter frames. See [Imaging System](#) and [Running with NINA](#).

Feasibility — Whether a target and filter can actually be imaged on a night: high enough (above your minimum altitude or custom horizon, set on Imaging System > **Horizon**), far enough from the moon, both at once, and for a long enough unbroken stretch. **Tonight's Feasibility** on Schedule > Nightly > Setup reports it without running a plan. See [Planning a night](#) and [Troubleshooting](#).

Greedy — A planner that fills the night strictly by your **priority order**: the highest-ranked target that can be imaged and still needs time goes first. Deterministic and instant. See [Planning a night](#).

Knee — The point on the Pareto front that balances utilization and quality most evenly — where giving up a little of one stops buying much of the other. The plugin's **Balanced** plan selection takes it. See [Running with NINA](#).

Master — A master calibration frame (dark, flat or bias) built by combining many individual calibration frames. The Processing tab's **Library** lists a rig's masters and matches them to your lights. See [Processing](#).

Minimum imaging time — The shortest stretch worth starting a target for (**Min imaging time** on each target card). A target whose feasible windows are all shorter than this may get no time at all. See [Targets](#).

Moon avoidance profile — A per-filter rule for how far from the moon a filter may image. The required separation peaks at full moon, falls off over the days either side, and can relax as the moon sinks toward the horizon, so narrowband and broadband need not share one crude cutoff. A hard constraint, never traded off. Defined on Imaging System > **Moon**. See [Imaging System](#).

Mosaic — A target made of several panels, each its own pointing with its own progress, grouped under one mosaic name. Panels share one priority rank. See [Targets](#).

Pareto front — The set of plans the optimizer returns, each one the best it found for its particular balance of **Utilization** and **Quality** — no plan on the front beats another on both. Drawn as the chart on Schedule > Nightly > Plan, where you pick the one you want. Called the GA front in older versions. See [Getting started](#) and [Planning a night](#).

Plan variant — One named plan in the Editor: built by hand, duplicated from another, taken from a point on the Pareto front, or generated by Greedy, Proportional or Dispatch. Every variant is plotted on the same chart as the Pareto front. Saved beside the config. See [Managing plans in the Editor](#) and [Files and data](#).

Priority order — The order you drag targets into on the Targets page. It drives the Greedy, Proportional and Dispatch planners and seeds the optimizer with one priority-ordered plan, but the optimizer's scoring ignores it.

Reset to plan order clears it. See [Targets](#).

Proportional — A planner that splits each night's darkness equally between the targets that can use it, so long-running projects advance together; priority order decides who goes first. See [Planning a night](#).

Quality — The second planning objective: how good the conditions were while exposures were taken, from airmass blended with how well the plan honors your filter-preference rules. Higher is better. See [Planning a night](#).

Readout mode — The camera readout mode for a filter row (the **Readout** column on a target's exposure plan). Left blank, the camera's own default is used. See [Imaging System](#) and [Targets](#).

Replan — Re-solving the rest of a night already under way, with what has been captured so far and a live weather reading — the NINA plugin's **Replan** instruction, usually fired by a weather trigger or after an unsafe spell. It uses the optimizer, and by default leaves a hand-built or generated plan alone. See [Running with NINA](#).

Schedule — The rail page holding the roadmap and tonight's planning, in two groups: **Seasonal (Setup, Plan)** and **Nightly (Setup, Plan)**, with the **Editor** as a chip inside Plan). See [Schedule](#).

Utilization — The first planning objective: how much of the night ends up as exposures. It blends the share of the night's darkness spent exposing with how much of each target's own imageable window gets used. Higher is better. See [Planning a night](#).