Guido API Summary

Succinct overview of the Guido library API, highlighting the most important modules, responsibilities, and project goals. Latin folder names preserve historical context; English function names provide clarity. The library is dependency-free.

Goals

- Unify calendar, chant corpora, notation parsing, mode analysis, tuning, and MIDI.
- Keep modules small, dependency-free, and composable.
- Preserve Latin naming for folders; use English functions for clarity.

calendarium

- Compute liturgical celebration for a date.
- Function: getCelebration(date, { forma }) → { feastId, title, rank, season }.

cantatorium

- Access chant corpora (GR, LA, LH, LU).
- Functions: search({ office, mode, incipit }), get(id).
- Records stay verbatim; labels derived via constants.

cantus

- Assemble chants for a celebration using calendarium + cantatorium.
- Handles rubrical selection (Alleluia vs Tract, commemorations).
- Function: buildPropers(celebration, { source }) → { propers, ordinary }.

notatio

- Parse GABC into tokens, notes, and lyrics with prosody.
- Apply rubrical markings (Solesmes via `solemnes`).
- Function: parse(gabc, { applyRubrics, prosody }) → { notes, tokens, lyrics }.

modus

- Analyze mode, final, tenor using modes.json + Guidonian hand.
- Function: analyzeMode(parsed) → { mode, final, tenor, confidence }.

temperamentum

- Map pitch classes to tuned frequencies (Just, Meantone, Pythagorean).
- Function: freq(pc, { temperament, a4 }) → Hz.

midi

- Convert parsed notes to MIDI events.
- Function: toMidi(parsed, { tempo }) → events[].

schola

- Provide analytics (ambitus, cadence, ictus density).
- Function: metrics(parsed) → { … }.

Happy Path

- date → celebration → chants → parsed notes → mode analysis → tuned frequencies → MIDI.