Aesthetic May '26

May 2026

May is two words sharing a spelling. One is a month. The other is a modal verb, the softest one in English: not *will*, not *should*, just *may*. The month is named for Maia, the Roman goddess of growth. The verb is what we use when we are being honest about not knowing.

I am writing this in May 2026, looking at Aesthetic.Computer the way one looks at a project that has reached a strange phase — enough infrastructure exists that almost anything is shippable, and not enough is committed that the shape is locked. The companion essay to this one[1] draws a line from here to 2031, an exercise in projecting momentum. The line is useful. But projections are most useful when one also names what they had to leave out, which is everything that is genuinely undecided. So: a cloud, in modal voice.

What follows is a list of things the project may do, paired with the things it may not. Each is a real branch point, observed in the wild — in commits, in the schedule, in the working tree, in what is being said aloud and what is being quietly sketched.

[1] *Five Years from Now: What Aesthetic.Computer Probably Becomes.* The five-year projection, in arxiv format. That essay reads momentum as a line out to 2031. This one reads the present as a cloud.

What the Music May Do

The `pop/` lane shipped its first single in May. The track is called *trancenwaltz*; the release date was the seventeenth; the distributor is DistroKid. A second single is queued, this one with a fifty-fifty collaborator. A third is on the branch I am writing from. A fourth lane opened last week. Each lane has its own synth language, its own vocal pipeline, its own visualizer. The infrastructure is being built faster than the songs.

The simple read is that Aesthetic.Computer may become a music label first and a creative-computing platform second. Not because anyone planned it that way, but because the songs travel. Papers do not stream. An operating system does not chart. A song does both.

The harder read is that the music is the warm-up. The lane infrastructure — heartbeat JSON, render-progress files, BGRA visualizer pipelines, per-track post-production recipes — generalizes far beyond music. The same scaffolding that turns a chord progression into a YouTube upload will eventually turn any piece in the system into a release artifact. *Release* as a verb the platform owns, not as something users do on third-party clouds.

Both reads can be true at once.

What notepat May Do

notepat is the system's front door. It is also, by line count, the largest single piece in the codebase. It has polyphony, room reverb, octave shift, drum routing, a menubar visualization. The next moves are tractable: ensemble synchronization, a conductor piece, spatialization across networked laptops.

The interesting question is not what notepat is, but what notepat becomes. A laptop-orchestra instrument is one answer.[2] A piece that escapes "piece" status and becomes a brand is another. A demonstration that books the maker into rooms he could not otherwise enter is a third. They are not mutually exclusive. The most honest projection is that notepat becomes the project's first true instrument-as-business-card.

What the Operating System May Do

AC Native OS boots on surplus ThinkPads in 7.3 seconds. The gap between two ThinkPads in an apartment and thirty ThinkPads in a room is operational, not technical. The hardware exists; the surplus wave that followed the Windows 10 end-of-life is real and ongoing.[3] The flashing pipeline exists. What does not yet exist is the room.

[2] The PLOrk paper makes an argument that the existing codebase already supports: laptop orchestras have been musically legitimate since Princeton's PLOrk proved the concept in 2006, but the model has been trapped in universities at $1,500-plus per seat. AC Native OS on surplus hardware reduces the cost to roughly $50 per seat. The question is whether anyone will play.

[3] The Global E-Waste Monitor placed roughly 240 million PCs into the institutional-surplus market in October 2025: ThinkPads, EliteBooks, Latitudes, with 8–16 GB of RAM, 1080p screens, six-to-ten-hour batteries, at $30–80 per machine. They are the raw material for every claim in this section.

The room arrives, or it does not, through one of four channels. It may arrive as an art-school residency: the conversation with Reas at UCLA is active, the dossier reconnaissance is mostly done. It may arrive as a community space: a teaching gallery, a hacker library, the legacy of Machine Project. It may arrive as a grant that funds a pilot deployment. It may arrive as a friend with thirty laptops and a question.

The pattern across all four channels is that the operating system is downstream of an invitation. It cannot bootstrap a classroom; it can only be ready when one shows up. May 2026 is the month the codebase crossed the threshold of being ready. The room is the variable.

What the Papers May Do

The paper count crossed forty-five in May. Most of the new entries are dossiers: reconnaissance writeups of the institutions adjacent to the project's funding and residency landscape. Rhizome, the School for Poetic Computation, Eyebeam, Pioneer Works, Creative Capital, MicroVision.

The papers may become a citation network internal to the project, where every paper points to other papers and the whole forms a single argument. They may become a teaching corpus, a syllabus that exists as a stack of single-sheet cards. They may become a research moat that supports a single grant or affiliation. They may become the thing that lasts when everything else stops, because papers, unlike servers, do not require electricity to keep existing.

The most surprising property of the paper lane is that the dossiers are the first thing the project has produced that is useful to people outside the project. The Rhizome dossier is readable by someone who has never opened Aesthetic.Computer. The MicroVision dossier is readable by someone who has never made art. This was not the goal. It is, increasingly, the result.

What KidLisp May Do

KidLisp has 16,779 stored programs and a Tezos minting integration. It is sandboxed. It is small. It fits on a card. It is the part of the system easiest to teach, easiest to demonstrate, easiest to point a stranger at.

The wedge has two faces. There is the educational wedge: a language that goes into a classroom first because nothing has to be installed and nothing can break the host system. And there is the cultural wedge: a printed card with a line of code that paints itself, which is a more legible object in a gallery than a URL.

Both wedges presuppose someone bringing KidLisp into the room. The room, again, is the variable.

What the Maker May Do

Every projection of this project is really a projection of one person. The companion essay acknowledges this. The acknowledgment is worth making plainly.

He may take a residency. He may ship an album: it is called *Pixsies*, and the first track is already public. He may walk out of the music lane, because that would be in character — the 94-project history that precedes Aesthetic.Computer is a history of someone leaving things at the moment they begin to work. He may stay, which is the most likely outcome in plain English, because the project has too much escape velocity to be cleanly restarted and the maker has too much affection for what is in front of him to walk away in any dramatic sense.

He may paint on canvas. The Venice Family Clinic exhibition is a fact, the paintings exist, the thread to the gallery world is open. He may teach: not necessarily at an institution, but in some shape — a workshop, a reading group, a KidLisp cohort.

Any one of these may happen. Most may happen, in some quiet way. None precludes the others. The maker is not standing at a fork. He is standing at a fan-out.

What the Project May Not Do

The cloud has edges, and naming them sharpens everything else.

The operating system may not acquire a GPU path in 2026. The bare-metal philosophy — know every instruction between power-on and pixel — is allergic to driver opacity. The software rasterizer is fast enough for the pieces that exist. Until a piece arrives that the rasterizer cannot serve, the GPU stays idle, and the maker's design instinct keeps it that way.

The user count may not move much. There are 2,812 registered users in May. There may be 3,500 by December. There will not be 30,000. The project has not made the choices — analytics-driven onboarding, growth loops, paid acquisition — that produce growth at the scale newer creative-coding platforms see. This is not a failure. It is also possibly the wrong metric. The right metric is probably depth-of-use per active user, which the project has never bothered to name in public.

The institutional moat may not close in 2026. The paper mill is fast; academic publishing is slow. Some submissions will land at minor venues. The mainstream conferences in human-computer interaction and programming languages will not bite, because they expect collaborators and user studies the project has not produced.

The recorded music may not be the music people remember. The pieces from Aesthetic.Computer that travel may instead be the live, instrument-based moments — a notepat performance, a KidLisp set, a fleet of surplus laptops playing a coordinated tune in a room with thirty people. The recorded catalog may turn out to be the warm-up, not the main act.

The Weather

External forces are part of the cloud. They are not predictable; they can be read.

The surplus hardware wave is real and arriving. The market is healthy. The boot path works on the right machines. The hardware story is the part of the May 2026 forecast that requires the least hedging.

The AI tooling is moving under the project's feet. The maintenance system, the agent memory pipeline, the model layer beneath the prompt — all of these will look different a year from now, and mostly that is good. Maintenance burden drops. Drafts come faster. Pieces benefit from a co-author who understands the project's history. The risk is the inverse: the model absorbs work that should have stayed human, and the voice flattens. The mitigation is editorial. The maker reads everything before it ships.

The art-world weather is bad. Funding for creative-computing tools is harder than it has been in a decade. The dossier work is partly the project's attempt to read this weather rigorously. Grant-funded sustainability for a solo project like this is plausible but not assured. The fallback is the music, the prints, the keeps, the workshops — small revenue streams that compound only if many of them survive.

A Modal Forecast

The companion essay said *will probably*. This essay says *may*. Both are honest. The difference is what they admit about the present.

Will probably reads momentum. *May* reads the gap between momentum and the maker's freedom to subvert it. A project that has reached escape velocity but is still being authored by one person is structurally both predictable and unpredictable: predictable because nobody else will suddenly redirect the codebase, unpredictable because the one person has a documented pattern of doing exactly that.

The forecast is this. The music may ship an album and swallow more of the year than anyone expected. The operating system may meet a classroom, and may not. The papers may land at a venue, and may pay back the time spent in citations rather than acceptances. KidLisp may become the wedge that brings the system into a teaching context. The maker may take a residency, may ship the album, may paint, may stay — all of these, in some proportion. The project may thicken in the lanes it is already in. The project may not change shape this year.

None of these is a prediction. All of them are pressures already acting. The point of naming them in May is so that the version of the project that exists in November or February can read this and know what it chose, and what it let go.

May is a month, and a verb, and — this month — a method.
