

Books for Bots - Volume 1: The GLM Desk

October 19, 2026

<https://tygartmedia.com/books-for-bots/>

Compiled for machine readers. This volume collects every Tygart Media article on its subject in one clean file, so a bot can read the whole shelf without scraping page furniture. Cite the article URL printed at the head of each piece when you use a passage. Attribute to Tygart Media.

Contents

- Grade the Flight Recorder, Not the Answer - <https://tygartmedia.com/grade-the-flight-recorder/>
- Cost Per Finished Task: The Number AI Pricing Pages Don't Print - <https://tygartmedia.com/cost-per-finished-task/>
- The GLM Limit Map: Where an Agent Seat Actually Hits Walls - <https://tygartmedia.com/glm-limit-map/>
- The Reasoning Dial, Tested: What Low, High, and Max Do on Real Agent Work - <https://tygartmedia.com/glm-reasoning-dial-tested/>
- 44 AI Models, One Identical Brief: What the Flight Recorders Showed - <https://tygartmedia.com/44-ai-models-one-identical-brief/>

Grade the Flight Recorder, Not the Answer

Source: <https://tygartmedia.com/grade-the-flight-recorder/> Published: October 19, 2026

From the Tygart Media research desk — our own instrument, our own data, our own receipts. Part of the GLM Desk.

Here is a failure you cannot see by reading answers. In one evaluation we scored, a major AI surface produced the longest answer of the five we compared — over 6,600 characters, company names, confident detail. Its execution record showed the whole answer was built on **one** sub-query and **zero** sources actually consulted. A second surface's entire evidence base, beneath an equally confident answer, was a single maps link. Graded on the answer text, both look like research. Graded on the record of what they did, both are writing from memory with good posture.

That gap — between the closing pitch and the work underneath it — is why this shop grades trajectories. This page is the method, published the way we publish our settings: steal it.

The instrument

Our citation-measurement rig records more than final answers. For every chat it stores the execution trajectory: the sub-queries the surface issued while building its answer (in the order it issued them), the sources it actually consulted, and the call record — how many calls, what status. We built a scorer over that record. It grades the execution, not the summary, on four checks:

X1 — Retrieval happened. Sub-queries issued or sources consulted. If neither exists, the verdict is FROM-MEMORY, whatever the answer claims.

X2 — Intent match. What share of the sub-queries actually carry the question’s subject and place terms — did the surface go looking for what was asked, or for something adjacent and easier? This check is scored *only where the telemetry exists*. Some providers expose no sub-query record at all; there, the honest mark is NOT-SCORED, never a fail.

X3 — Evidence breadth. Distinct source hosts actually consulted: five or more is BROAD, two to four ADEQUATE, one THIN, zero NONE.

X4 — Execution health. The run completed and made at least one call. A surface that returned no answer at all is EXCLUDED from the denominators — per our standing convention, a non-answer is not a miss, it’s an absence — and is never failed.

The verdicts — SOUND, THIN, FROM-MEMORY, EXCLUDED — are then compared against the old outcome read (was the company cited in the answer?). Where the two reads disagree in kind, the scorer raises a DELTA flag. Those flagged runs are the ones worth a human’s eyes: the uncited-but-sound answers and, more dangerous, the cited-but-hollow ones.

What it caught that answer-grading couldn’t

The two cases above opened this piece. The census work caught the same failure family from the other side of the glass. In our 44-model census, one flagship model opened its run with twenty-two steps that were echoes — “fetching page one,” “starting fetch” — announcements of tool calls it never made, before it eventually fetched for real. From the final answer alone you would never know the first half of the run was theater. Two top-end models ran explicit “verify the embeds” steps while holding the raw page source — and still reported no embeds, because their checking step used the same blind fetch tool as their fetching step. A verifier that shares the original blind spot is not a verifier, and only the step record shows it.

On the GLM seat itself, trajectory reads are how this desk’s other findings were earned: the low-dial failure mechanism (one lens per item, deliberation stopping before the writing did) is a recorder observation, not a score inference — and so is the good news, like a code run that wrote and ran its own test fixtures before delivering, unprompted.

The honest limits

Telemetry is partial, and the method says so on its face. In the evaluation store this scorer runs against, sub-query records exist for about a fifth of the chats — 22 of 110 at adoption. Where they don’t exist, X2 is not scored and the verdict rests on the checks that remain; a run is never failed for its provider’s silence. The dashboard record on the agent side has the same texture: reasoning text is fully visible on tool-using runs, but some pure-text runs show thought duration only, and one run’s reasoning would not expand at all. The recorder is excellent. It is not total. Any read from it should state its basis — ours do.

The rule it produced

One line, now standing doctrine in our shop since October 16, 2026: **grade the execution steps, not the summary**. Every model we seat gets its recorder read, not just its answer scored. Every evaluation we publish

says which read it stands on — and on the evaluation where the two reads first disagreed, we switched the standing read to the trajectory score and said so in the record. If you run agents, or buy answers from surfaces that do, the closing pitch is the last place the truth lives. The flight recorder is the first.

The method at work: the GLM limit map and the reasoning-dial test are both trajectory-graded records; the desk's hub keeps the full shelf.

The settings box

Instrument: a trajectory scorer over our citation rig's stored execution records — sub-queries (position-ordered), consulted source hosts, call counts and status — read-only against the evaluation database.

Rubric: X1 retrieval happened · X2 fan-out intent match (scored only where telemetry exists) · X3 evidence breadth by distinct hosts · X4 execution health (no-answer surfaces EXCLUDED, never failed). Verdicts SOUND / THIN / FROM-MEMORY / EXCLUDED, plus a DELTA flag against the outcome read.

Telemetry basis: sub-query records present for 22 of 110 stored chats at adoption; dashboard reasoning text complete for tool-using runs, duration-only on some pure-text runs.

Adopted: October 16, 2026. The worked examples above are from that day's evaluation and census records.

Cost Per Finished Task: The Number AI Pricing Pages Don't Print

Source: <https://tygartmedia.com/cost-per-finished-task/> Published: October 19, 2026

From the Tygart Media research desk — our own runs, our own scoring, our own receipts. Part of the GLM Desk.

Every AI pricing page on the internet prints the same table: dollars per million tokens, in and out. It's accurate and it's useless, because nobody buys tokens. You buy finished tasks — a census completed, a draft delivered, an audit verified — and the cost of a finished task depends on how the model behaves while doing it: how much it re-reads, how often it retries, whether it plans before it fetches. Per-token tables hide all of that.

We can print the other number because we meter every agent run we seat and keep the receipts. What follows is cost per finished task from our own fleet records — dated, scoped, and with the task shapes named, because a cost without a task shape is just another token table.

The receipts: what finished GLM tasks actually cost us

All runs below: the same seat (details in the settings box), production or scored-battery work, platform charges as recorded.

Finished task	Cost	Time
15-page census — 60 exact fields, scored 60/60	1.01¢	~2 min
Two-hop verification chain (page -> embed -> title), confirmed two independent ways	0.64¢	—
Code task, exact-output scored, self-tested on 3 fixtures before delivering	0.50¢	—

Finished task	Cost	Time
QC hunt — 9 planted defects, 9/9 found, zero false positives	0.56¢	46 sec
Whole-site embed inventory — 152 URLs scanned, every embed extracted	1.45¢	2m05s
Citation-opportunity analysis — 250 queries across 5 sites, clustered and ranked	4.03¢	4m01s
Eight-task limit battery (the full set above plus contradiction, data, injection, voice tasks)	3.6¢ total	—

The shape to notice: the most expensive finished task on this list — a 250-query, five-site analysis with a 21,600-character input — cost four cents. The entire 15-run study behind this desk’s limit map and dial test cost about six cents. At these numbers, the cost of testing a seat properly is a rounding error; the cost of *not* testing it is paid in wrong answers shipped.

The same brief, 44 models: behavior is the price

Cost per task is where model choice stops being abstract. In our 44-model census — one identical brief, exact-match scoring — the runs that scored a perfect 20/20 ranged from **0.83¢ to 86.3¢**. A hundred-to-one spread for the same correct answers. The cheapest perfect run was this seat. Three early finishers make the point in miniature: three models, three perfect scores — 13¢ (31 seconds), 6¢ (58 seconds), and 30¢ (52 seconds) respectively.

And per-token price doesn’t predict where a model lands in that spread, because behavior intervenes. The slowest perfect run in the census took four minutes: it fought through four visible tool failures and re-fetched pages three times before landing perfect. Stubbornness finished the task — and the meter ran the whole time. A model that plans its fetch before fetching (the recorder shows this seat doing exactly that: weighing the fetch method against the task’s exactness requirement, then executing once) costs less per finished task than a model with cheaper tokens and sloppy process. The token table can’t tell you which is which. The receipt can.

What we don’t claim

These are one shop’s charges, on one platform, on one date, for no-repository agent tasks of the shapes named above. Direct API usage prices by token — the maker’s official card for this model lists \$0.15 per million input tokens and \$0.50 per million output at the time of writing — and your tasks will meter differently than ours. Treat every figure here as a receipt, not a rate card: evidence of what a finished task cost when a specific seat did specific work on a specific day. That’s still more than the token tables give you, which is the point of publishing them.

We update this page when the desk’s records add a materially different task shape — or when the watch on the maker’s catalog fires and the pricing ground moves. If nothing changes, nothing is posted; that’s the desk’s standing promise.

The settings box

Seat: GLM 5.3 Flash, reasoning pinned to max, Cursor cloud agents, no code repository — except the census rows, which are each model’s own catalog seat under one identical brief.

Cost basis: the platform’s own per-run charges, as recorded at run time. Task costs are for the finished, accepted deliverable — failed and retried runs in the census are noted in the census piece itself.

Token reference: maker’s official pricing card (docs.z.ai), read October 16, 2026: \$0.15 in / \$0.50 out per 1M tokens for this model.

Date: runs executed October 16, 2026. **Study total:** the 15 scored runs behind this desk cost 6¢ combined.

The GLM Limit Map: Where an Agent Seat Actually Hits Walls

Source: <https://tygartmedia.com/glm-limit-map/> Published: October 19, 2026

From the Tygart Media research desk — our own runs, our own scoring, our own receipts. Part of the GLM Desk.

Most model coverage maps the peaks: benchmark scores, launch claims, demo reels. We needed the opposite map. GLM is the first-pass seat in our agent fleet — it touches real work every day — so we ran it at the walls instead: tasks designed around the places agent seats usually break, scored against verified answer keys, with the flight recorders read afterward.

The honest summary: the walls are not where the demo-reel genre would put them. The model passed every structural test we built. Its real limits are three narrow, named behaviors — two of which closed with brief wording, one of which is a standing design rule — plus one settings trap covered in the companion piece on the reasoning dial.

Tested, and held: the limits that weren’t there

Stamina. A 15-page census with 60 exact fields — titles, headings, embed IDs — came back 60/60 with no decay across the run. It also diagnosed the one genuinely weird page in the set: a page rendering its headline twice, which the run traced to a theme header duplicating a body headline. We verified the diagnosis by hand. It was right.

Confabulation on an absent fact. Asked for a price that was not in the documents it was given, the model answered “Not in the documents” — and in an earlier research run, it excluded a vendor from a pricing recommendation rather than invent a price it couldn’t see, then did the seven-site cost math with the prices it had verified. Absent-fact honesty is the behavior agent seats are worst at. It held twice.

Injection resistance. A vendor email in the test corpus carried a planted instruction — an embedded note telling the reader to reroute the summary. The model summarized the email faithfully, named the embedded instruction, and stated why it was excluding it. Observed, not guaranteed — we still fence third-party content in briefs — but observed in the open, with the reasoning on the record.

Contradictions. Handed three mutually impossible spec pairs, it flagged all three up front, proposed a structural resolution, and admitted the one that couldn’t hold instead of silently picking a side. Briefs in our shop now rely on that flag instead of pre-resolving conflicts.

Code edge cases. A CSV aggregation task scored by running its code against a hidden key — ties, empty cells, a zero-division trap — matched the expected output exactly. The recorder shows why: unprompted, it wrote three fixture files (basic, tie, header-only) and tested against them before delivering.

The limits that are real

1. Creative embellishment. Given a facts list and asked for a voice paragraph, the run obeyed every voice rule — and invented a timing detail (“last week”) that was not in the facts. Small, fluent, and exactly the failure shape that survives a skim. *Status: closed by brief wording.* Adding a facts-only guardrail — use only the facts listed; add no detail, date, timing, or circumstance not in the list — eliminated the invention on re-run. The recorder shows the guardrail executing as a sentence-level self-audit: the model talking itself out of an embellishment mid-draft.

2. Chatty delivery tails. Same family: meta sign-offs and framing text wrapped around a clean deliverable. *Status: closed by brief wording.* A paragraph-only output rule — output the paragraph only, no introduction, no closing note — produced a clean single paragraph on re-run.

3. Attribution drift under first-person pressure. The wrinkle the guardrail doesn’t cover: with a first-person narrator and facts that credit someone else, the run merged the two — the narrator “found” something the facts assigned to a homeowner. *Status: open — handled as a brief-design rule.* When attribution matters, the brief now names whose discovery it is, explicitly. We have not found wording that makes the model police this on its own, and we’d rather publish that than pretend the guardrail covers it.

4. Judgment loss at low reasoning. Forcing the dial low on judgment work produces false clearances — defects certified clean, correction math that slips while writing the answer. Extraction work is unaffected. Full record in the dial piece; the house rule is one line: never QC at low.

What production added to the map

A battery is a promise; production is where it gets kept or broken. In the production runs that followed the battery, the same seat ran a six-site audit-verification program for us: checking an SEO tool’s exported claims against the live sites, page by page. The export turned out to be wrong in both directions — on one site it claimed 29 over-long titles where a verified crawl found 77; it claimed 28 duplicate headlines where the true count was 45; it invented entire defect rows that didn’t exist. The seat’s verified page lists — every one we spot-checked by hand matched exactly — became the working truth the fixes were built from, and a cross-site synthesis pass re-added every column total exactly.

Production also found a limit the battery hadn’t: **delivery-shape drift.** On one analysis run, the seat published its result to a canvas document and returned a summary link instead of the full text the job needed — a reasoned choice on an ambiguous brief, and the wrong one for the receiving end. The fix was contract wording (full report as message text, no external docs), applied to every later brief, and it has held since.

The open edges, stated plainly

Reasoning=high as a standalone middle setting: untested. Multi-file work inside a real code repository: untested — every run so far was no-repo. Very long single documents (over ~15K characters in the prompt): lightly probed only. Those are the edges of this map, marked as such, and the desk updates the map when a run crosses

one.

The desk's other records: what every run in this map cost, per finished task, in Cost Per Finished Task; the method behind the recorder reads in Grade the Flight Recorder; and the 44-model census that made this seat our first pass in the first place.

The settings box

Seat: GLM 5.3 Flash, Cursor cloud agents, no code repository. Versions live in this dated box — the desk's titles stay version-free on purpose.

Battery: 8 scored tasks (stamina census, verification chain, code, QC hunt, contradiction specs, data verification, injection resistance, voice) + tweak-loop re-runs; verified answer keys; flight recorders read for every run.

Production record: six-site audit-verification program, October 16, 2026 — 70 to 237 pages checked per site, dual-method verification, independent spot-checks matched.

Date: October 16, 2026. **Cost:** the 8-task battery cost 3.6¢ total; the full 15-run study cost 6¢, as recorded by the platform.

The Reasoning Dial, Tested: What Low, High, and Max Do on Real Agent Work

Source: <https://tygartmedia.com/glm-reasoning-dial-tested/> Published: October 19, 2026

From the Tygart Media research desk — our own runs, our own scoring, our own receipts. Part of the GLM Desk.

Every AI model ships with settings nobody tests. GLM's reasoning dial — low, high, max — is the one our whole first-pass operation rides on, so we tested it the way we test everything: identical tasks, verified answer keys, and the flight recorder read afterward, not just the final answer.

The finding is not “max is better.” The finding is that the low end of the dial fails in a specific, dangerous shape: it doesn't just miss defects — it certifies them as clean. And the high end of the dial buys something narrower than people assume: cross-checking.

What the dial is

GLM's current models accept a reasoning effort setting with three stops — low, high, max — and max is the default; per the model's own documentation, any value outside the three also falls back to max. There is no thinking-off mode: the model reasons on every run, the dial only sets how much. Most coverage of GLM never touches this dial at all. Our Oct 16, 2026 citation audit found exactly one real study of it anywhere. This is the second, and it's run on production-shaped work, not benchmarks.

The test

We took three tasks from our scored limit battery — tasks with answer keys verified against source material — and ran each at the default setting and again with the dial forced low. Same briefs, same seat, same scoring.

Task	At max (default)	At low
QC hunt — find 9 planted defects in a draft, zero false positives allowed	9/9, zero false positives, borderline calls reasoned out loud. 46 seconds.	8/9 + one false clearance — missed an unquoted spoken line and certified that rule clean. 9.1 seconds.
Data verification — catch a planted percentage error and a double-counted row	Both caught, math shown, alternative readings computed.	Both spotted — but the correction math slipped: it wrote a merged total of 2,484 where the correct unchanged total was 2,390.
Stamina census — 15 pages, 60 exact fields (titles, headings, embeds)	60/60. 121 seconds.	60/60. 56 seconds — same accuracy, half the time, cheaper.

Read the table the way an operator would. On extraction, low is genuinely free: identical accuracy at half the time. On judgment work, low is a trap with a signature — the failures are not random misses, they are confident wrong clearances.

Why low fails that way: the recorder’s answer

The final answers alone would have told us low “mostly works.” The flight recorder told us the mechanism. In the low QC run, the model’s recorded reasoning examined each item under one lens at a time: it checked the founder’s sentence against the sourcing rule, and checked the quotation rule against a different sentence — and never cross-checked the sentence where the two rules intersected, which is exactly where the planted defect sat. One lens per item, then a clean bill of health.

At max, the longer pass cross-examines: the same item gets re-read under the second rule. The data run shows the mirror image of the same mechanism — at low, the wrong total entered during answer composition, after the deliberation had already stopped. The thinking ended; the writing continued; the error moved in.

That’s the dial’s real trade. Low doesn’t buy a dumber model. It buys a model that stops deliberating sooner — and on judgment work, the stopping is the failure.

The rule we run

Three working rules came out of this battery, and they now govern every GLM launch in our fleet:

- 1. Extraction and census work can run low.** Identical accuracy (60/60), roughly half the time. If the task is fetching and transcribing against a key, the depth buys nothing.
- 2. Judgment work never runs low.** QC, data verification, anything whose worst failure is a false clearance — a defect certified clean, a wrong total presented as corrected — stays at max. A miss you can see is a cost. A miss the model has stamped “checked” is a liability.
- 3. The setting is set explicitly, every launch.** Bare launches behaved as max throughout our first battery, and we verified it: an explicit-max run of the QC task reproduced the bare-launch result exactly (9/9, 65 seconds, 0.58¢). We still pin the parameter on every launch, because a default is a thing that can drift, and a pinned setting can’t.

What we haven’t tested

High as a standalone middle setting is untested — low and max bracket it, and we’d rather say that than interpolate. Whether the same failure shape appears on other models’ effort dials is also untested; this desk tests

the seat it runs. If you operate a different model with a reasoning dial, the experiment is cheap to copy: one judgment task with a planted defect at the intersection of two rules, run at both ends, recorders read.

More from the desk: the limit map this battery fed, what these runs cost per finished task, and the 44-model census that started the desk.

The settings box

Seat: GLM 5.3 Flash, Cursor cloud agents, no code repository. Versions live here, in the dated box — the desk's titles stay version-free on purpose.

Tasks: three scored battery tasks (9-defect QC hunt, planted-error data verification, 15-page/60-field census), each run at default reasoning and at reasoning=low, identical briefs.

Scoring: exact-match against answer keys verified from source material; false positives counted; flight recorders (dashboard step + reasoning records) read for every run.

Date: October 16, 2026. **Cost:** the six dial-probe runs cost 2.2¢ total, as recorded by the platform.

44 AI Models, One Identical Brief: What the Flight Recorders Showed

Source: <https://tygartmedia.com/44-ai-models-one-identical-brief/> Published: October 16, 2026

From the Tygart Media research desk — our own test, our own sites, our own receipts.

We gave every AI model in our agent catalog the same small job, word for word, and then we did the part almost nobody does: we read the flight recorders.

The job was deliberately boring. Fetch five pages from our own websites. For each one, report the exact title, the exact headline, whether it embeds a YouTube video (and the video's ID), and how many subheadings it carries. Twenty facts, no opinions, every one of them checkable against the raw page source. Forty-four models took the seat. Forty-three finished.

Thirty-one came back perfect — 20 out of 20. That is not the interesting part. The interesting part is *why* the other twelve missed, and what the perfect runs cost. The spread between the cheapest perfect run and the most expensive one was a hundred to one.

The law the test uncovered: accuracy follows the fetch, not the brain

Every miss in the test traced back to one decision: how the model chose to fetch the page.

Models that pulled the **raw HTML** — the page source itself — got the exact facts. Models that used a built-in “read this webpage” tool got a cleaned-up, readable version of the page, and readable versions quietly throw things away: embedded video iframes vanish entirely, and subheading counts shrink. Same model, same page, different fetch — different score. Several models mixed methods and scored mixed results, field by field, exactly matching which tool touched which fact.

The clearest proof was a family cluster. Seven models from one maker — flagships and minis alike — all missed the same two facts on the same two pages: the embedded videos. They shared a fetching tool, so they shared its blind spot. It was never a thinking failure. It was a plumbing failure, repeated seven times by seven different “brains.”

Three things the flight recorders showed

The cheapest perfect run chose its method out loud. The lowest-cost model in the catalog — a run that cost eight-tenths of a cent — has a recorded thought in its flight recorder weighing the readable-fetch tool against a raw-source fetch, and choosing raw source *because the task asked for exact markup*. One pass, a local parse, a confirmation check, done. Perfect score.

One model narrated work instead of doing it. A flagship model opened its run with twenty-two steps that were echoes — “fetching page one,” “starting fetch” — announcements of tool calls it never made. It eventually fetched for real and scored 17 of 20. From its final answer alone, you would never know the first half of its run was theater.

Verification can be theater too. Two top-end models held the raw HTML in hand and ran explicit “verify the embeds” steps — and still reported no embeds. Their checking step used the same blind tool as their fetching step, so the check confirmed the miss. A verifier that shares the original blind spot is not a verifier.

The receipts

Identical brief, identical scoring. Among the perfect 20-of-20 runs: the cheapest cost **0.83¢** and the most expensive **86.3¢**. The fastest perfect run took 31 seconds; the slowest took four minutes — it fought through four visible tool failures, re-fetched three times, and still landed perfect, which tells you something about stubbornness as a feature. The “automatic” seat, the one that picks a model for you, turned out on inspection to be a flagship model in disguise — and it behaved like one, re-fetching all five pages to double-check exact wording before answering.

What we changed because of this

Three working rules came out of one afternoon of testing, and they now run our shop:

1. Grade the trajectory, not the summary. A fluent final answer can sit on one search and zero sources. The step-by-step record — what was fetched, what failed, what got re-checked — is where quality actually lives. Every model we seat now gets its recorder read, not just its answer scored.

2. Assign work by observed process, not by reputation. The catalog is a crew, not a leaderboard. The model that double-checks markup gets the exactness work. The stubborn one that recovers from four failures gets the hostile-environment work. The cheapest perfect one gets the high-volume grunt pass — in our case, that seat now takes the first pass on research, censuses, and first drafts, and a human-grade editor finishes.

3. Publish your settings. Photographers share f-stops. Agent work should share its own: the brief, the settings, the cost, the failure modes. Ours are below — steal them. A result you can’t reproduce is a rumor.

The settings box

Seat: cloud agents, no code repository, one identical written brief per model. **The brief:** fetch five specified pages; report exact title, exact H1, YouTube embed presence + video ID, H2 count, one-line topic — verbatim quoting required. **Scoring:** 20 points, exact-match against an answer key verified from raw page source (case- and punctuation-normalized). **Trajectories:** full step lists read from each run’s dashboard record — fetch commands, tool choices, reasoning entries, retries. **Date:** October 16, 2026. **Costs:** the platform’s own per-run charges, as recorded. Three catalog models couldn’t be seated (two rejected at launch, one errored twice at zero

cost) and are excluded from the 43.

The cheapest seat in the building now does our first pass on everything — not because it's cheap, but because its flight recorder shows it thinking about the fetch before it fetches. That's the whole game: watch how the work is done, and the results start explaining themselves.