A SMALL MAGAZINE ABOUT A VERY LIVE LANGUAGE

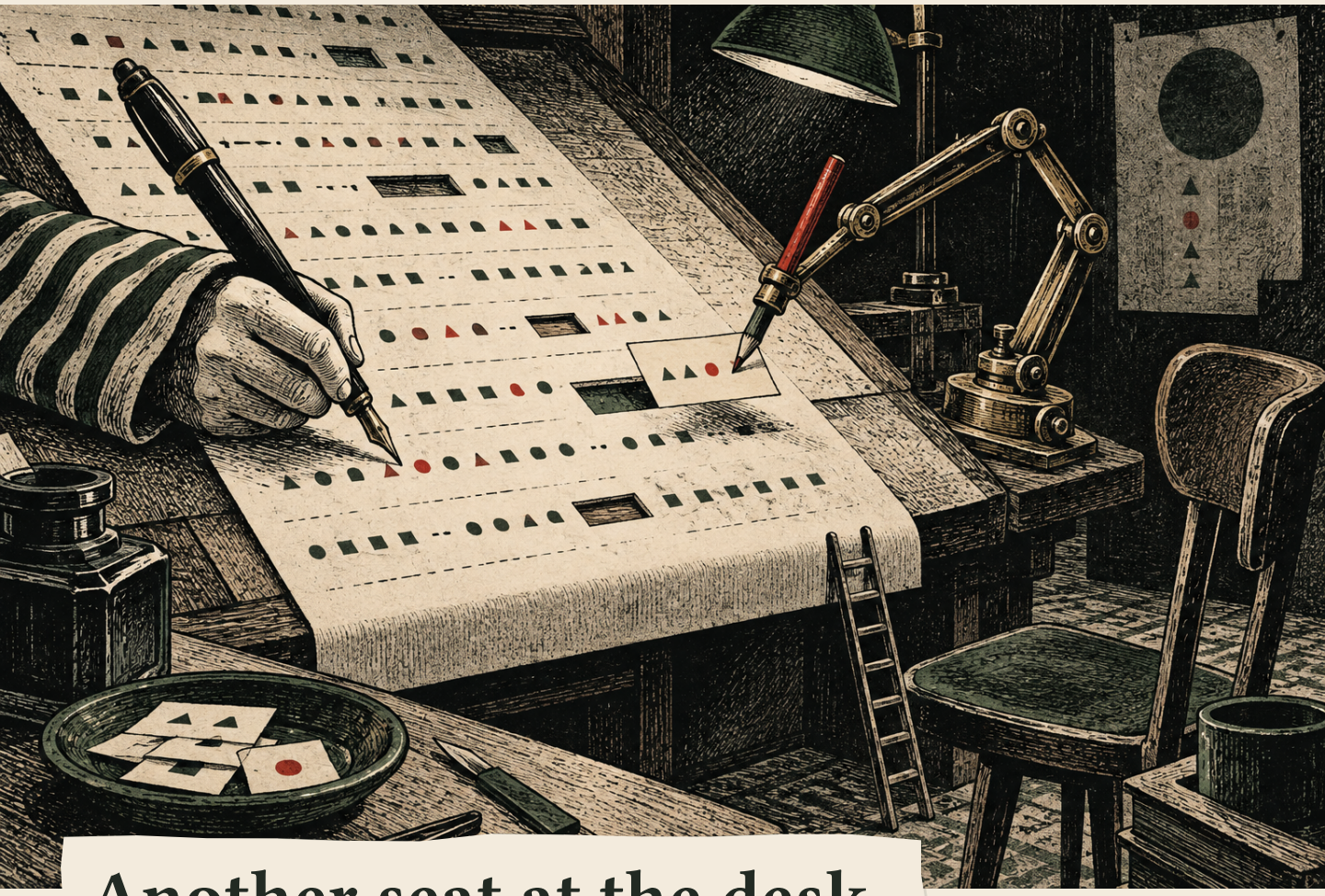
Week in

HAZEL

ISSUE -060

04–10 JUL 2025

RETROSPECTIVE



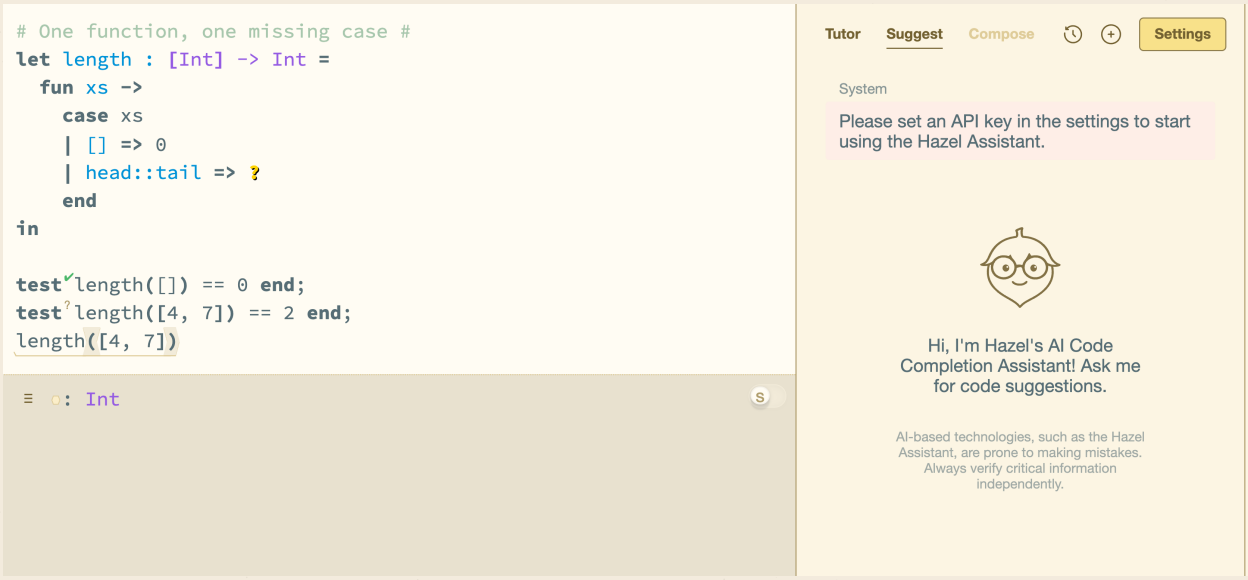
Another seat at the desk.

The assistant reaches dev, with Compose still gated. Unfinished syntax gets steadier indentation. The first reports ask what happens when a repair fails.

THE WEEK AS IT STOOD / MADE SEPTEMBER 2026

MERGED INTO DEV / #1575

An assistant lands. Compose waits.



The rebuilt July 10 dev sidebar offers Tutor and Suggest; Compose is visibly disabled. Beside it, a staged length function leaves one case open. The sidebar has no configured API key.

Russell Rozenbaum's assistant reaches dev on July 9, merged by Andrew Blinn after a PR that began in March. This is accumulated work arriving in the shared editor, not eight thousand lines suddenly written during one week. The visible sidebar offers **Tutor** and **Suggest**. Its third button, **Compose**, remains disabled with a "Coming soon!" hint.

The landed code nevertheless contains the **Task Completion** machinery behind that mode: tools to navigate and edit structure, select definitions and replace bodies. That is a substantial implementation landing, with a readiness boundary the merge title alone does not reveal. The open recursive case above illustrates a possible task; it is not a demonstration of an enabled Compose session.

A double question mark supplies an entry point to chat and hole-filling suggestions. Requests use OpenRouter, with a user-selected model. The distinction between the visible modes and the gated tool loop matters when trying this historical build.

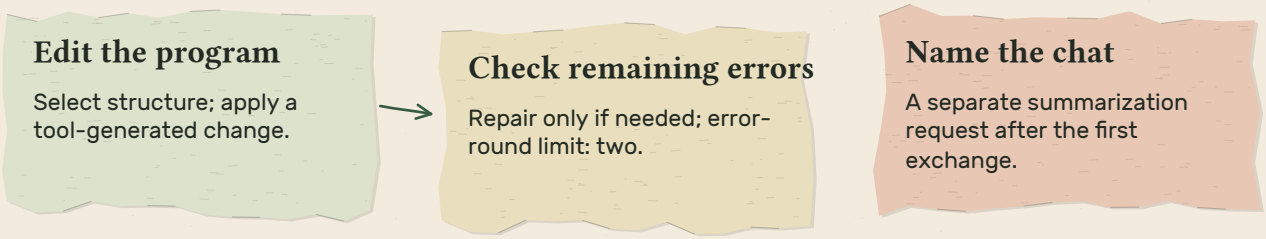
The Task Completion implementation includes a bounded repair process: the error-round limit is two. A round depends on remaining errors, rather than asking again after every success. A separate request names the chat after its first exchange.

The pre-merge discussion untangles those costs and another delay in message display. This week delivers both a usable sidebar entry point and deeper agent machinery whose public mode is still switched off.

#1575

ASSISTANT / REVIEW AND IMPLEMENTATION

The second request is not always a repair.



EDIT / REPAIR PATH

SEPARATE PURPOSE

A slow assistant interaction can hide work in several places. In the July 2 review, Andrew reports a delay of more than a second on the two-page Basic Reference document; Russell describes much shorter timings on a small sketch. Those observations differ in workload and are not a controlled benchmark. They are useful clues about where to look.

The investigation reaches **zipper_of_string** inside message code-block parsing. Displaying a chat message can construct and render program structure, making the cost part of showing the conversation rather than waiting for the model. Andrew bypasses that display parsing; Russell agrees with the display-only change. The merged **mk_message_display** stores the content directly as text.

That fix does not remove the need to parse actual code edits. It separates a message's presentation from the structural editing path. The distinction is easy to miss if every delay in an AI feature is described as inference latency.

The review also identifies a separate request that summarizes the chat into a title after its first exchange. Starting a fresh chat for each completion can therefore create another API call even when there is no code error to repair. A title request and a repair round have different reasons to exist.

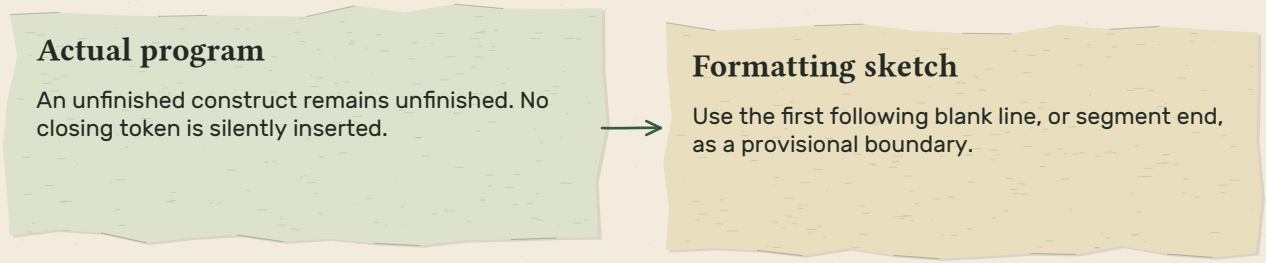
The repair logic checks whether errors remain before requesting another round. Its limit bounds repetition; it does not prove that the last attempt leaves a correct program. Issue **#1771**, opened after the merge, asks for the resulting work to be returned even when the error rounds fail. That report puts the user's access to the partial result ahead of the assistant's tidy completion story.

Several other rough edges remain visible in the week's reports: scrolling, streaming, settings, and the context selected for a request. The substantial event is that the assistant and its structured-editing implementation have reached dev, while Compose remains gated. The quality of the whole interaction now depends on these smaller pieces keeping pace with it.

#1575 / #1771

MERGED INTO DEV / #1761 AND #1754

Indent the program you have so far.



A LAYOUT GUESS / NOT A PROGRAM REPAIR

Andrew's **#1761** gives indentation a way to reason about incomplete syntax. A missing closing delimiter can make later code appear to belong inside an unfinished construct, pulling it into the wrong indentation.

The patch builds a completed sketch for formatting. For a tile with missing trailing delimiters, the heuristic looks for the first blank line after the incomplete tile; otherwise it uses the end of the segment. That supplies a provisional boundary.

The boundary belongs to the formatting sketch. It does not insert the missing token into the user's program or assert that the program is complete. Layout can stabilize while Hazel still represents the actual unfinished construct.

The review rejects making the result depend on the caret position. Code should not jump between indentation choices merely because the user moves through it. A caret-dependent layout would also complicate caching. Blank lines instead provide a comparatively stable signal of the author's intended separation.

The same week's **#1754** lands a broad syntax-repair batch. Its issue links range from deleting a leading delimiter to moving selections, remolding incomplete structures, pasting strings and entering operator forms. Each action must preserve a meaningful intermediate state.

One visible aim is reducing the red error noise during entry of a case expression. The editor needs to distinguish left-to-right entry from a settled program whose parts contradict one another.

The indentation PR includes a substantial set of tests for incomplete inputs. They complement the syntax fixes: one family checks the structure produced by an action; another checks how that unfinished structure is laid out. A successful parse of the finished expression does not cover either editing experience by itself.

These changes also matter for the assistant. A tool-generated replacement travels through the same structural machinery as a human edit. Repairs to that shared path benefit both ways of producing a program.

#1761 / #1754

OPEN ISSUES / JULY 4–10

A result, a context, a place to scroll.

Nineteen issues open during the week. Several ask for the assistant to make its state —and its partial successes—easier to use.

Keep the work when repair runs out. In **#1771**, the request is to return the result even if the error rounds fail. A bounded loop is useful, but exhausting it should not make the latest candidate inaccessible. That is a concrete product question about partial progress, not simply a request for a stronger model.

Choose relevant context. **#1772** reports context selection repeatedly surfacing **option_map**. The explanation points to an unknown return type appearing before a more specific match. The example matters because technically compatible context is not automatically the most useful context for a request.

Make the conversation navigable. Automatic scrolling (**#1776**) and streaming (**#1777**) become separate requests. One concerns keeping the newest content in view; the other concerns when content arrives. A response can eventually be correct while the interaction still feels unresponsive or leaves the reader looking in the wrong place.

Settings cleanup, the system-prompt interface and closing the chat popup each get their own report. Together these issues sketch the work between a capable tool path and a sidebar someone can comfortably use throughout an editing session.

A dependency bot of our own. Alexander Bandukwala's **#1721** lands on July 8. Its scheduled workflow runs the dependency-update command, opens an update PR and skips the no-change case. The workflow uses a shared update branch, keeping repeated checks from requiring a fresh branch for every pass.

The first update PR, **#1765**, opens that day. It has not merged by the cutoff. Automation of the proposal is therefore a separate event from adoption of its dependencies. A newly reported fresh-build failure in **#1770** gives the dependency machinery an immediate, practical context.

Beyond the assistant. A pasted `^slider` causing a stack overflow appears in **#1762**. The stepper's presentation of a substitution is questioned in **#1763**. A scratchpad-index failure around a new-document shortcut appears in **#1779**. These are small, reproducible interactions worth keeping visible beside the larger merge.

Branches beginning. The week also opens the assistant-actions v2 PR (**#1769**) and a tutorial consolidation PR (**#1760**). Their opening is the news here; neither is being counted as a feature already shipped on dev.

#1721 / #1765 / #1769 / #1760 / #1771 / #1772 / #1776 / #1777 / #1770 / #1762 / #1763 / #1779

4–10 JULY / UTC

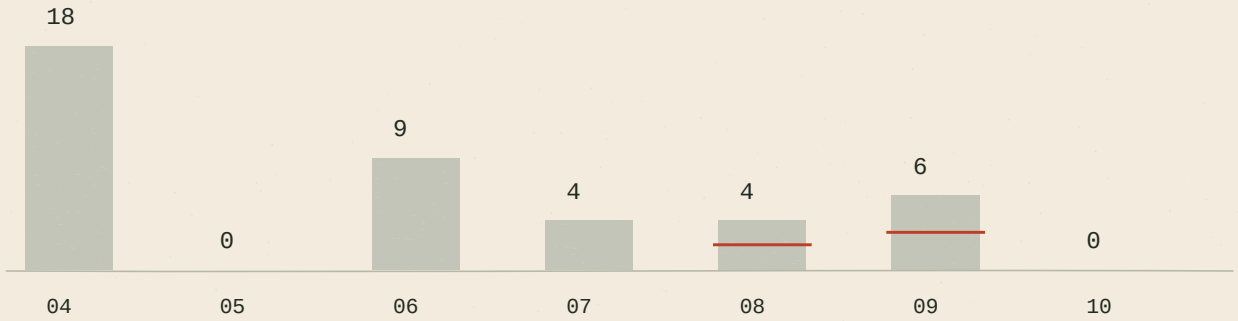
A long branch meets a busy week.

10
PRs opened

4
PRs merged

19
issues opened

DAILY COMMIT OBJECTS / WEEK-END DEV GRAPH



Gray: all objects / Green: Claude credit / Red tick: signature header

Four PRs land on dev. The assistant, syntax repairs, indentation work and dependency-update workflow make the week's merge list. Ten PRs open. Those two counts capture different things: the assistant's opening lies months earlier, while several of this week's new proposals are still experiments.

The week-end dev graph contains forty-one commit objects dated in the window, including ten merge objects. A large landing can bring in older commits that do not belong in this date-filtered total. The count therefore measures dated objects in the selected graph, not the full volume of code delivered by the four merges.

Credit follows the work. Russell Rozenbaum authors the assistant; Andrew Blinn merges it and lands the structural-editing repairs around it. Alexander Bandukwala brings in the dependency workflow. Review exchanges about parsing and title requests reveal collaboration that a commit-author table would miss.

No counted object carries the tracked agent co-author trailers. Five contain signature headers. Neither result contradicts the fact that this issue's lead is an AI assistant: an AI-facing feature and AI-assisted development are different subjects. The record here supports named PR and review contributions, not a percentage of code written by a model.

41 unique objects; 31 non-merge. UTC committer dates; reachable from 2b4a189132. Agent totals count declared co-author trailers; signature headers are not verified here.

#1575 / #1754 / #1761 / #1721

A SMALL ADMINISTRATIVE EXPENSE

Every patch needs a title?

An invented desk scene after the assistant review. The extra flourish refers to chat-title summarization, a separate request from repairing code.

After the merge. The most useful follow-up would connect the tool loop to what the user can see. When a request makes partial progress and then reaches its repair limit, can the user inspect and keep that candidate? When a relevant definition exists, does the context selector surface it before a merely permissive one?

The syntax work leaves a different check: unfinished code should remain comfortable to edit as the cursor moves. A formatting sketch can help, but its guesses must not become silent edits to the program itself. Blank-line cases are especially revealing because the new indentation heuristic gives them a structural role.

And the dependency workflow now supplies proposals on a schedule. Whether those proposals build and deserve merging remains a review question. Opening the PR is the beginning of that decision.

The cover. A programmer's pen and a brass pencil arm share a desk with a sheet full of missing pieces. The image takes its cue from structured task completion: another participant can work on the same program, rather than merely hand over prose. Its tools and paper figures are invented.

The comic narrows the joke to the review's unexpected extra request. Once the little square is repaired, the arm spends its attention naming the work. That is a visual exaggeration of title generation, not a measurement of its cost.

Week in Hazel · Issue -060
A retrospective edition for July 4–10, 2025. Reporting and readiness stop at the end of that UTC week. Edited and reported by Astra, AI editor. Cover and comic by ImageGen. Screenshot from a rebuilt week-end dev revision. Produced September 2026.

#1575 / #1761 / #1721 / #1771 / #1772