Harness Beats Hype: Test-First Agent Loops, Pi, and Monty

Coding Agents Alpha Tracker

2026-03-15

Harness Beats Hype: Test-First Agent Loops, Pi, and Monty

By Coding Agents Alpha Tracker • March 15, 2026

Simon Willison's test-first agent playbook was the clearest signal today, while Pi and Monty showed where serious users are pushing the harness layer: tighter context control, typed execution, and better review loops. This brief pulls out the concrete workflows, model-routing patterns, and repos worth stealing from.

TOP SIGNAL

Simon Willison published the clearest public playbook today for making coding agents less magical and more repeatable: start every session with the exact test command, tell the agent to use red-green TDD, then force a manual `curl` pass after the tests because green suites still miss real bugs [1]. The bigger cross-source takeaway: the wins are coming from harness discipline—tests, templates, rewinds, scoped workers, and sandboxes—not from giving one model unlimited rope [1, 2].

> "Tests are no longer even remotely optional." [1]

TOOLS & MODELS

- **Pi** — minimal system prompt, top-five benchmark leaderboard performance with only basic file/bash tools, and strong context controls. The real signal is model routing: Haiku for question extraction, Sonnet 4.6 for well-scoped workers, Codex for review; Armin says that level of control matters because hidden harness changes and context injections kept breaking his Claude Code workflows [2]
- **Monty + Pydantic AI** — typed host functions, built-in TY type checking before execution, and in-process execution measured in ~800ns hot

loops / single-digit microseconds. Samuel Colvin positions it as useful when a full sandbox is too slow or too awkward to self-host [3]

- **Claude Code + Gemini CLI + Codex** — Samuel mostly codes in Claude Code, uses Gemini CLI for fast whole-branch review reports, then points Claude Code at the report to implement fixes; Codex is a second reviewer when he wants a more agentic investigation [3]
- **OpenClaw** — next release adds `/btw`, a small but useful primitive: you can ask agents questions even while they are busy working. Docs are already up [4]

WORKFLOWS & TRICKS

- **Simon's default session loop**
  1. Tell the agent how to run tests (`uv run pytest`)
  2. Add: `use red-green TDD`
  3. After codegen, have it start the server in the background and exercise the API with `curl`
  4. If you want a readable audit trail, tell it to use Showboat so it writes a Markdown log of the manual test run [1]
- **Conformance-first implementation** — Simon's Datasette file-upload trick: ask the agent to build a test suite that passes against multiple reference implementations, then implement your own version against that shared behavior [1]
- **Seed the repo so agents copy the right things**
  - Use templates with tests, README, and CI
  - Keep at least a couple tests in your preferred style
  - Agents are extremely consistent at following existing patterns, so good scaffolding compounds [1]
- **Use sub-agents surgically, not as a feature factory**
  - Pi users keep 40-60% of context free by planning first, breaking work into todos, sending defined tasks to Sonnet 4.6 workers, then rewinding to a warm parent context for polish [2]
  - Armin's caution: sub-agents help with exploration and parallel search, but if you still read most of the code, swarms can just hand you too much to review [2]
- **Security hygiene that survives model churn**
  - Avoid the "lethal trifecta": private data + malicious instructions + an exfiltration path [1]
  - Containerization protects the host, but Armin says it does not solve secret exfiltration; Simon prefers Claude Code on the web when he wants the work contained off his laptop [2, 1]
  - Do not clone prod data to local laptops; generate mock users and edge cases instead [1]
- **Two small workflow unlocks**
  - Armin now routinely lets agents write small Python scripts instead of JavaScript because `uv run` made dependency handling simple enough

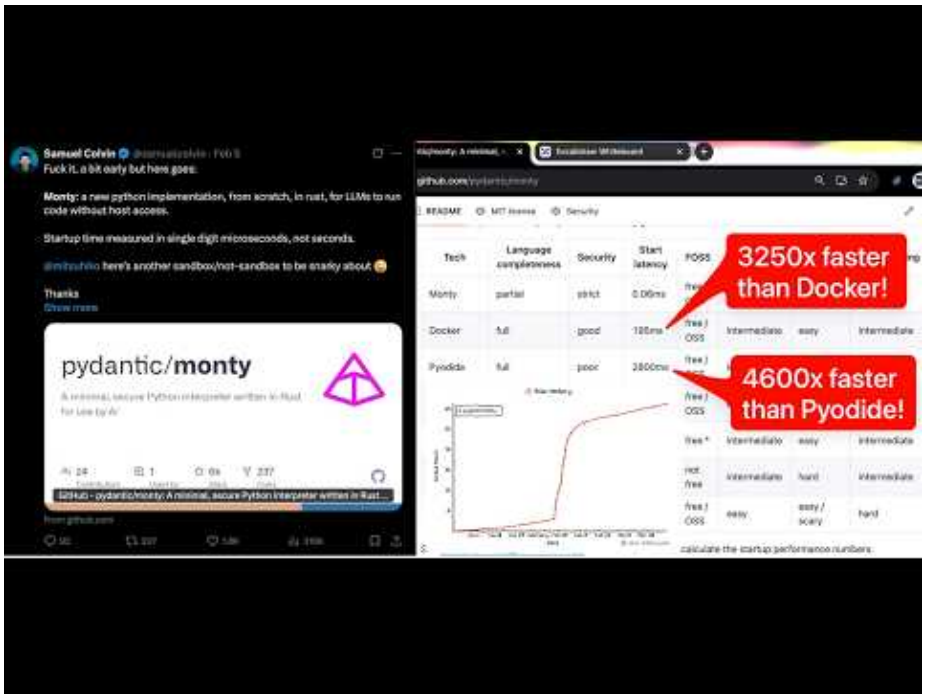
[5]
  - `git bisect` gets much easier to drive through an agent loop [6]

PEOPLE TO WATCH

- **Simon Willison** — dropped a quote-rich Pragmatic Summit fireside chat and notes; worth it for the TDD/manual validation/safety playbook and for his explicit rejection of "nobody reads code" workflows in security-sensitive contexts [7, 1]
- **Armin Ronacher** — high-signal because he keeps surfacing small workflow changes that actually matter: `uv run`, agent-friendly `git bisect`, and real `/autoresearch` usage on MiniJinja [5, 6, 8]
- **Samuel Colvin** — strongest current voice on type safety, constrained host functions, and mixing models for review vs execution [3]
- **Peter Steinberger** — worth following for OpenClaw tooling, but also for the framing: this is "agentic engineering," not sloppy vibe coding; you still need thinking, testing, debugging, and iteration [9, 4]
- **Dimitri** — useful counterweight to autonomy hype: hands-off codegen currently tops out around a couple thousand lines of standard code, and enterprise rollouts are likely to force a review-heavy phase first [10]

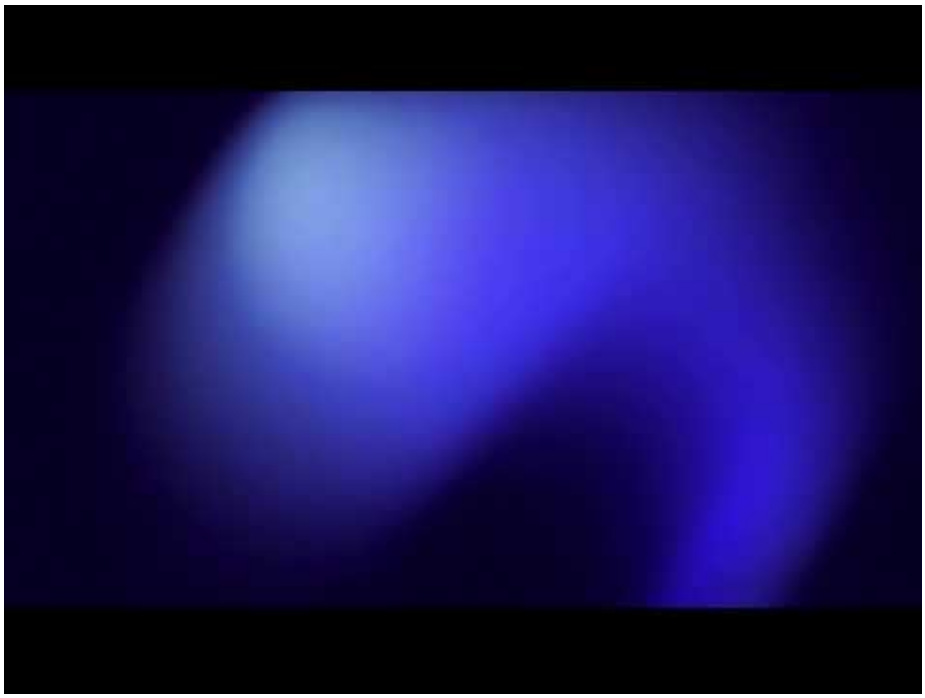
WATCH & LISTEN

- **11:23-13:08 — Latent Space / Samuel Colvin**: The cleanest explanation today of when coding agents jump from "a bit faster" to roughly **100x faster**: known internals, known API, easy tests, and no bikeshedding about the interface [3].



Monty: the ultrafast Python interpreter by Agents for Agents — Samuel Colvin, Pydantic (11:23)

- **65:20-68:23 — Pi AMA**: Armin's take on memory for coding agents is worth hearing in full: the codebase is the source of truth, and agentic search beats hauling around stale summaries [2].



Pi Day: AMA with Pi's Creator + Talks & Extensions Deep Dive (65:20)

- **27:45-29:56** — **TheStandup / Dimitri**: Useful reality check if your company is mandating AI use: the likely near-term outcome is a review-heavy workflow that many engineers will hate [10].

What's really going on with AI, Expert weighs in | TheStandup (27:45)

PROJECTS & REPOS

- **Pi extension stack** — Todos, Answer, screenshot/debug tooling, and patch-based multi-edit experiments are where the project feels differentiated right now [2]
- **`pi-autoresearch`** — now past the toy stage: Armin ran it overnight on MiniJinja, got many perf improvements, and is reviewing the resulting PRs one by one. Context: MiniJinja PR #884 [11, 8, 12]
- **Showboat** — Simon's new agent QA tool that turns manual test execution into a Markdown artifact you can actually inspect later [1]
- **`lossless-claw` + qmd memory plugin** — if OpenClaw's stock memory is weak for your use case, steipete is explicitly pointing people to these alternatives [13]

Editorial take: the durable edge right now is harness design, not raw model bravado—tests, context boundaries, and constrained execution keep showing up in every workflow that actually works. [1, 2, 3]

Sources

1. My fireside chat about agentic engineering at the Pragmatic Summit

2. Pi Day: AMA with Pi's Creator + Talks & Extensions Deep Dive
3. Monty: the ultrafast Python interpreter by Agents for Agents — Samuel Colvin, Pydantic
4. X post by @steipete
5. X post by @mitsuhiko
6. X post by @mitsuhiko
7. X post by @simonw
8. X post by @mitsuhiko
9. X post by @steipete
10. What's really going on with AI, Expert weighs in | TheStandup
11. X post by @tobi
12. X post by @mitsuhiko
13. X post by @steipete