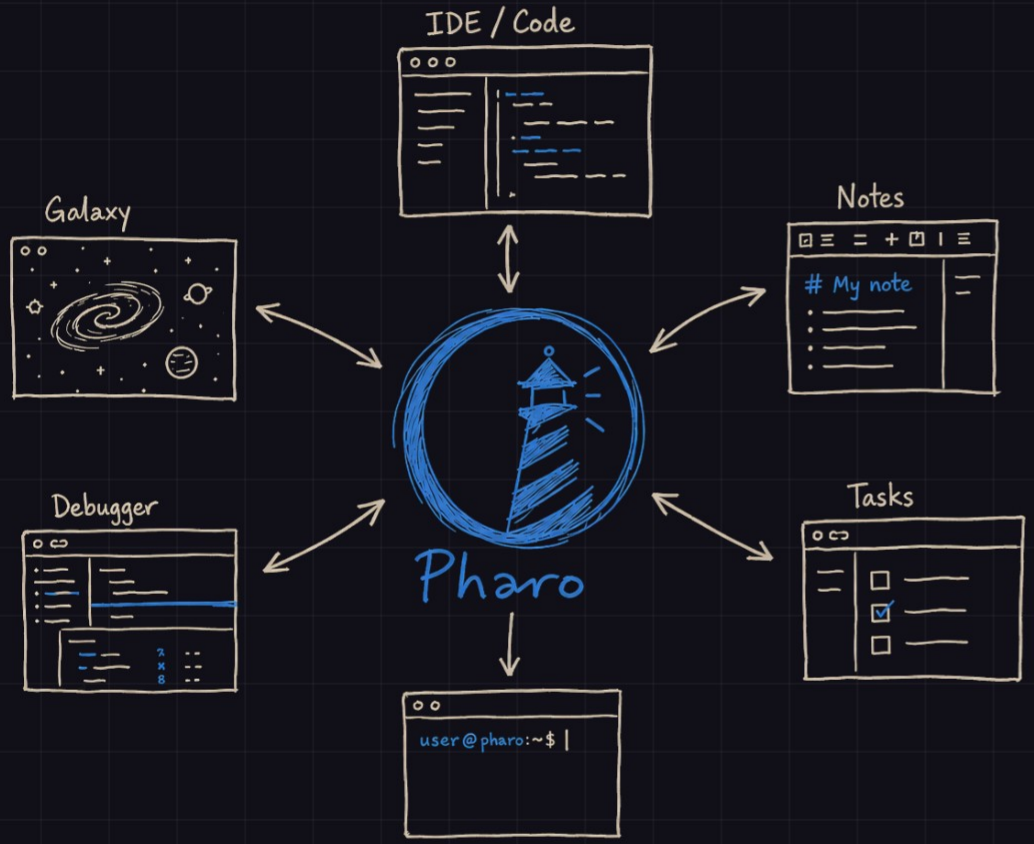


ESUG 2026 · PLOVDIV

Pharo on the Desktop

A tour of tools, friction, and feedback loops

Esteban Lorenzano

The desktop was declared dead

Again, and again, and again.

- applets
- PHP / JSP / web frameworks
- “my language to JavaScript”
- WebAssembly
- Electron everywhere

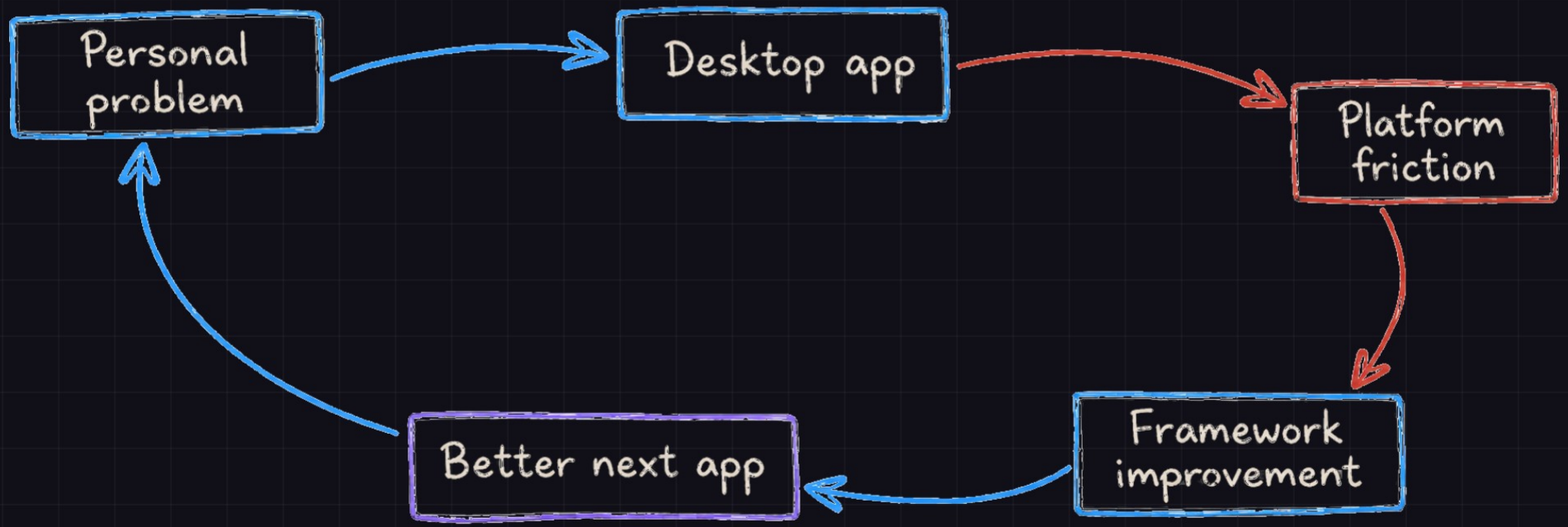
And yet...

We still build tools that need to be **fast**, **local**, **integrated**, and **shaped around real workflows**.

The question

→ Can Pharo be used not only to build Pharo, but to build desktop applications I genuinely want to use every day?

The feedback loop



SHOW ME THE APPS!

Context: “ELITE: DANGEROUS”

A game from the 80’s

- Explore and mine the galaxy.
- Get some combat.
- Procedurally generated

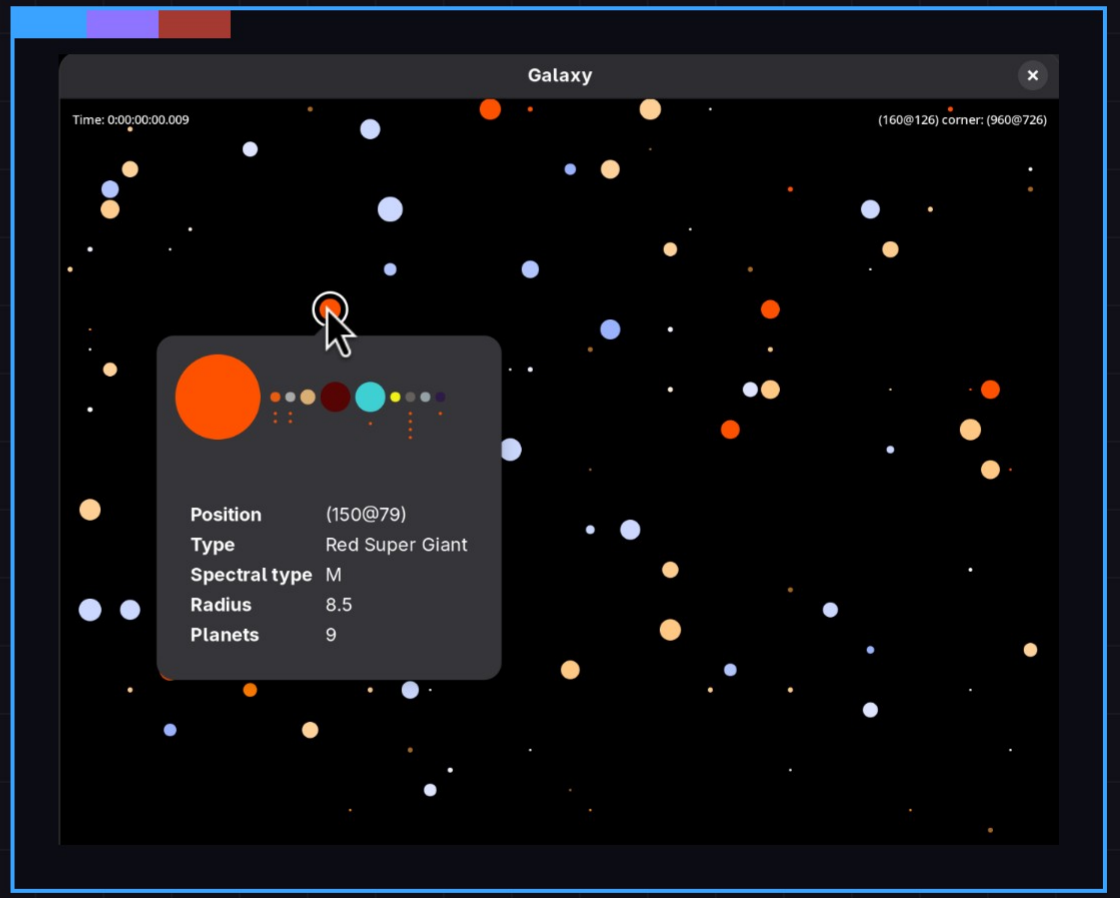
Let’s simulate our own galaxy…

Procedural Galaxy

STARTING POINT

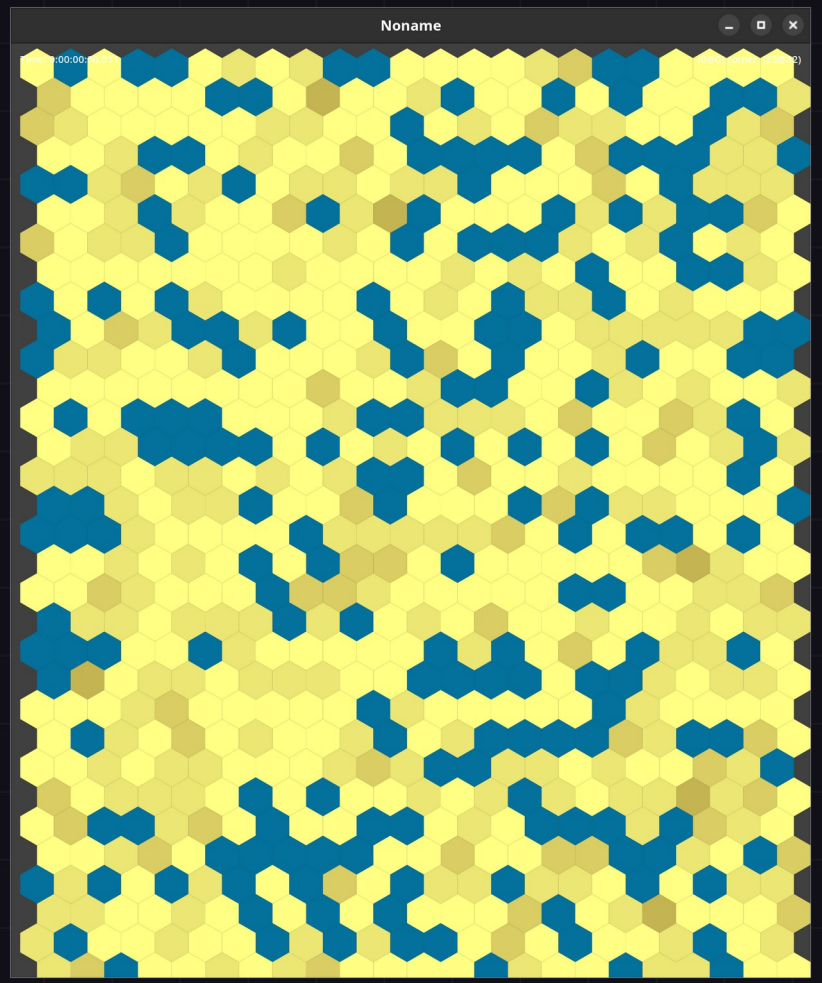
An infinite procedural galaxy:

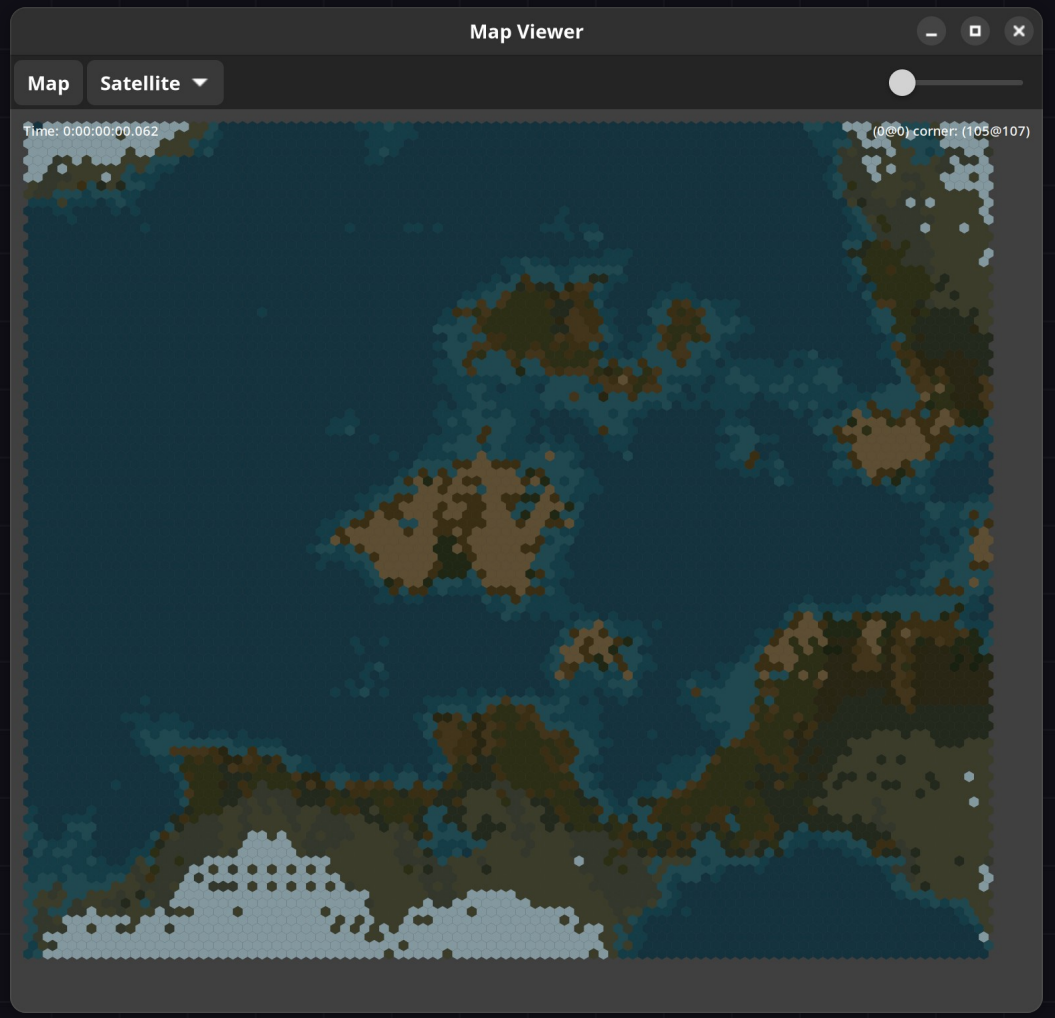
- stars, types, sizes
- planets and moons
- scrolling exploration
- hover interaction



And before you ask... yes, I also built the planets :)

(but I had to stop when real work called me...)

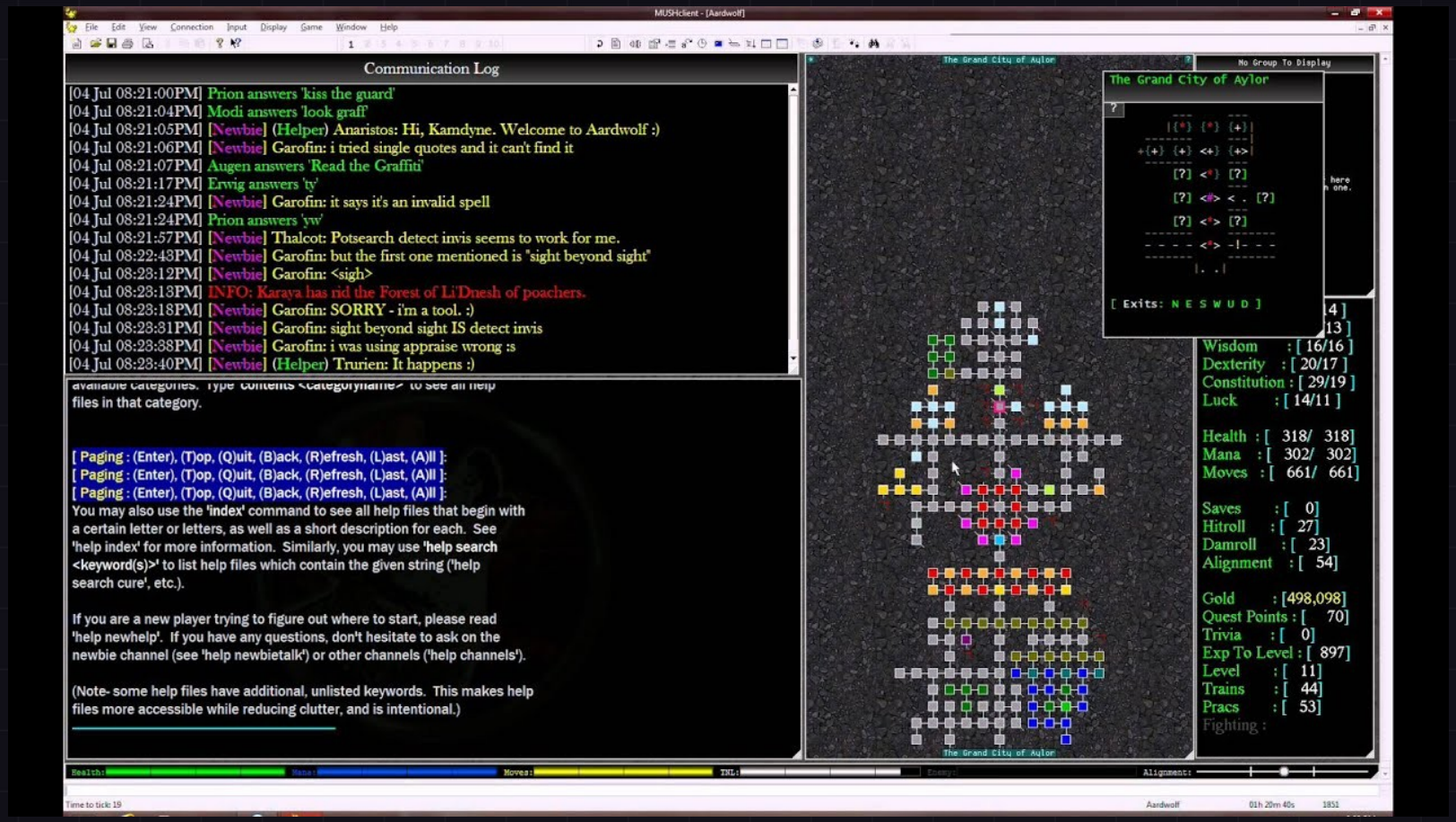




Context: “Multi User Dungeons”

Another game from the 80’s

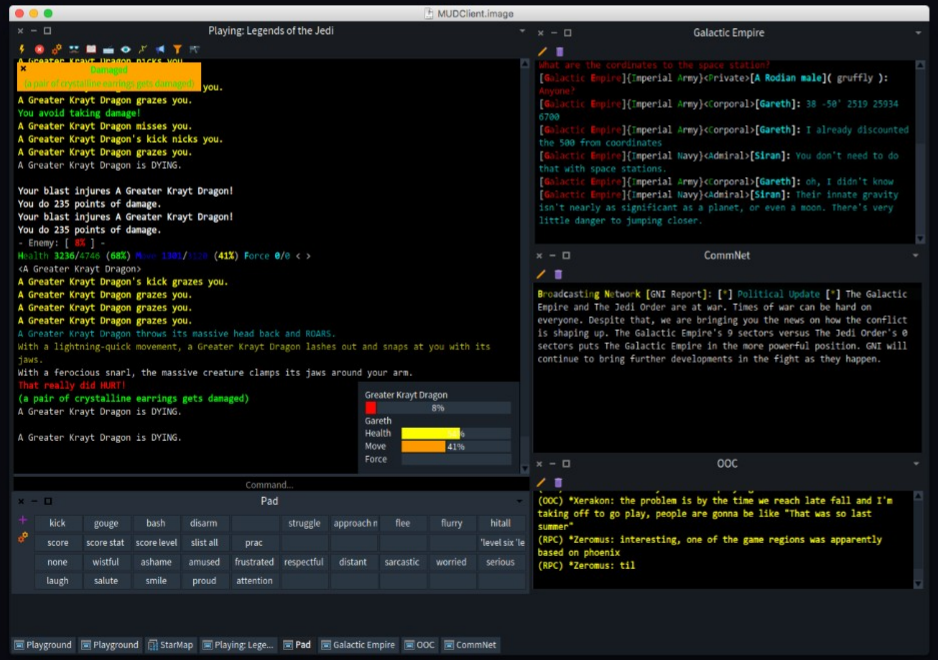
- Role playing games like a pro.
- Everything in a terminal !
- … and telnet protocols.

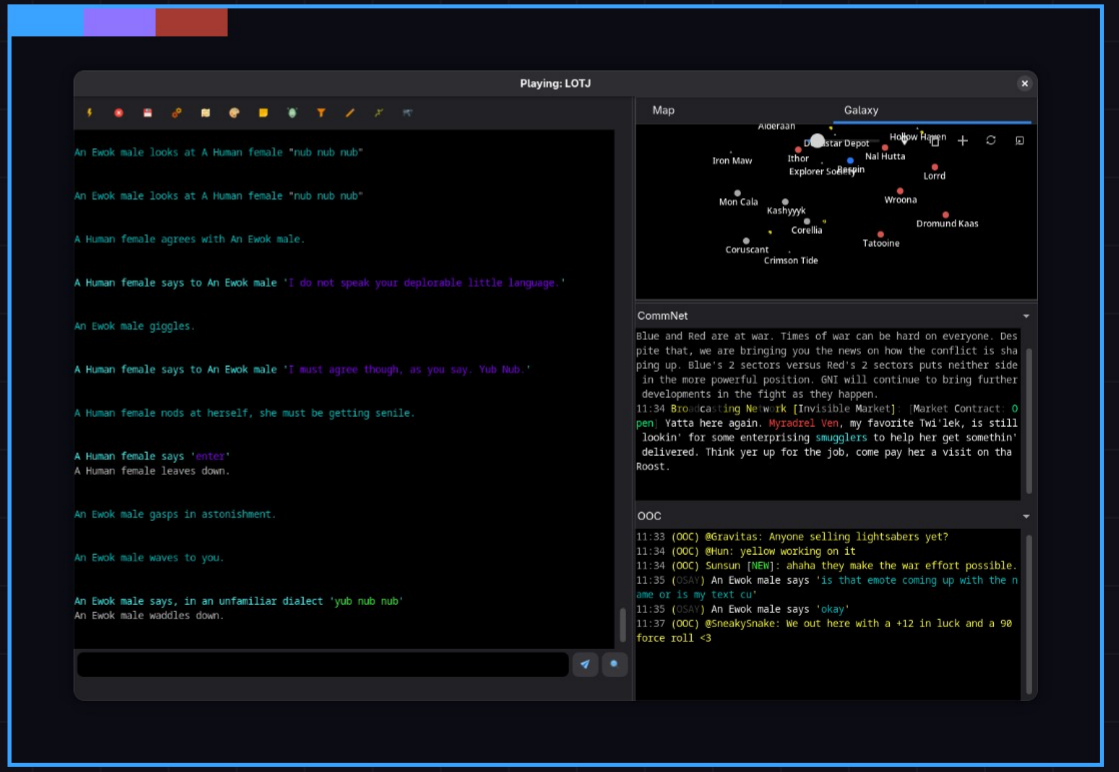


Let’s connect to a MUD and play…

MUDClient

OLD GAME, OLD PROJECT, NEW BACKEND

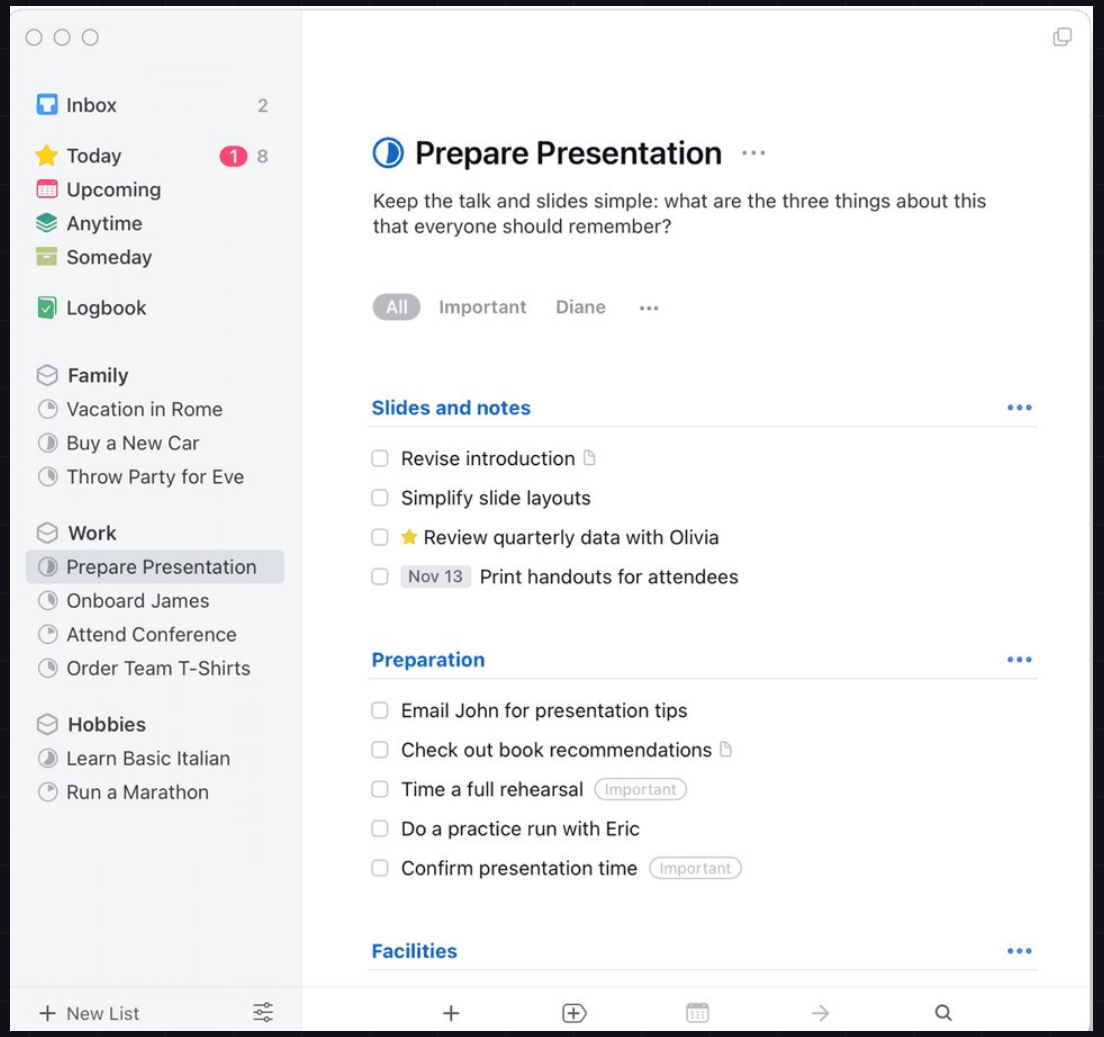




Context: “GTD and productivity in general”

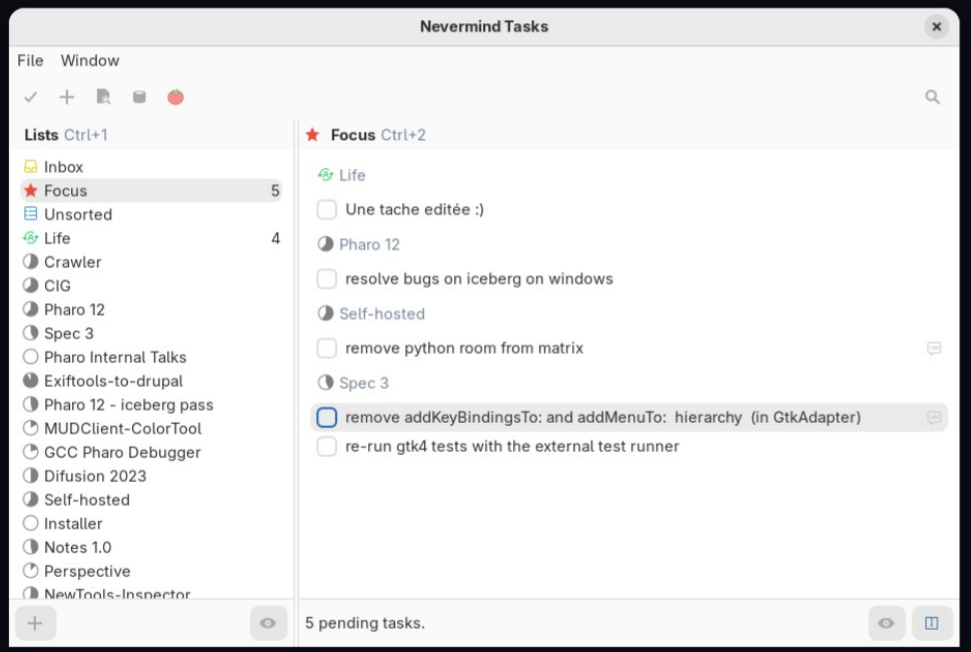
“Get Things Done”

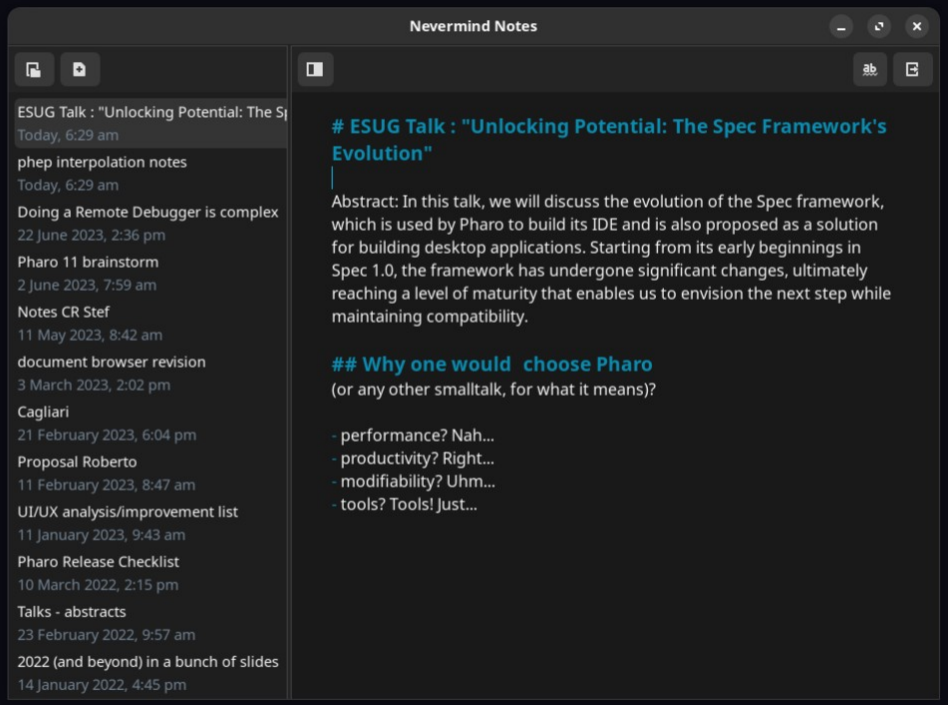
- A technique to avoid procastination (kind of…)
- Markdown notes
- Pomodoro



Let’s do our own…

Nevermind Tasks and Notes

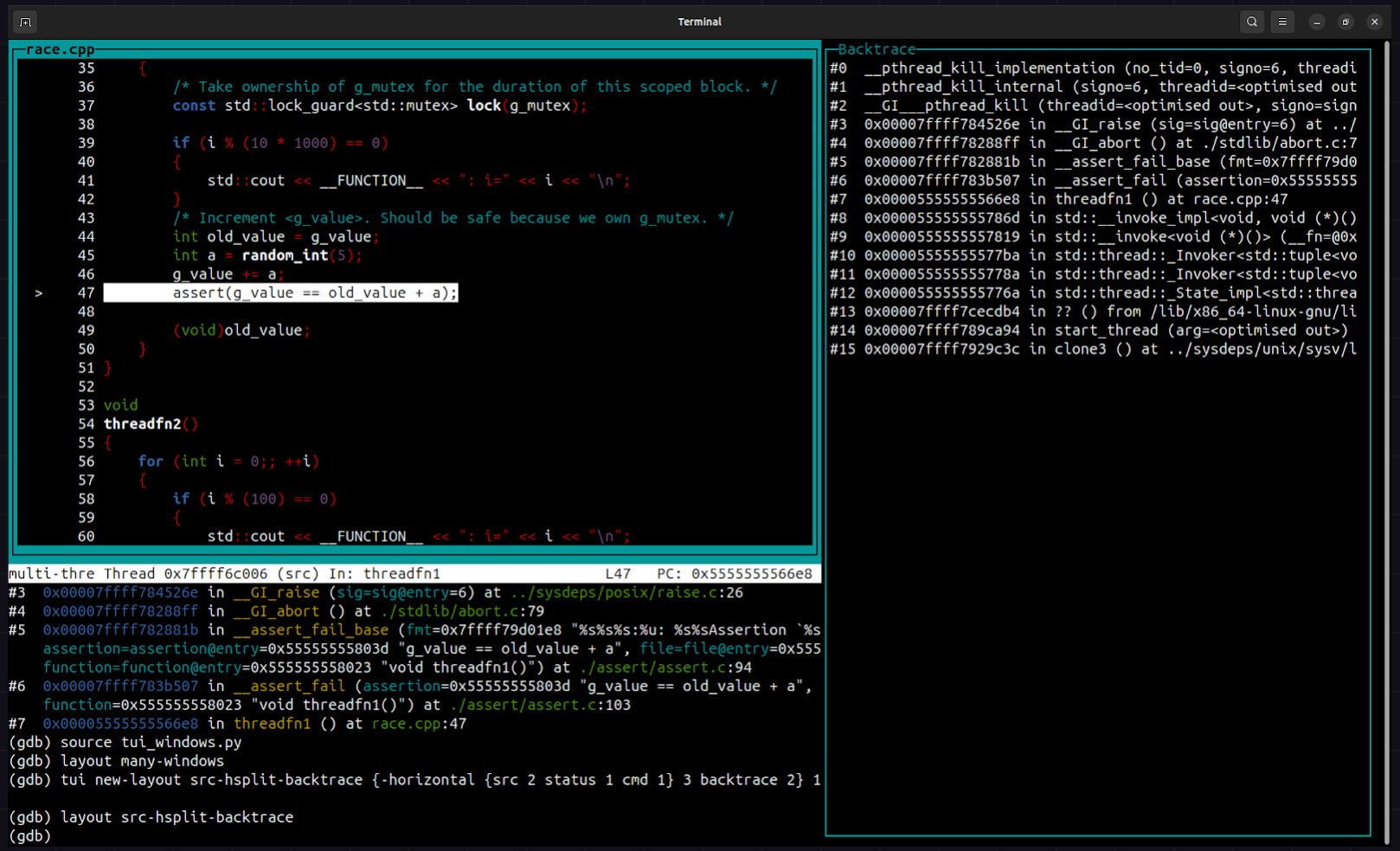




Context: “Application crashes”

… and when you use C libraries, you have C crashes

(This is the sexier thing you can get with gdb TUI)



You got the idea…
Let’s do our own !

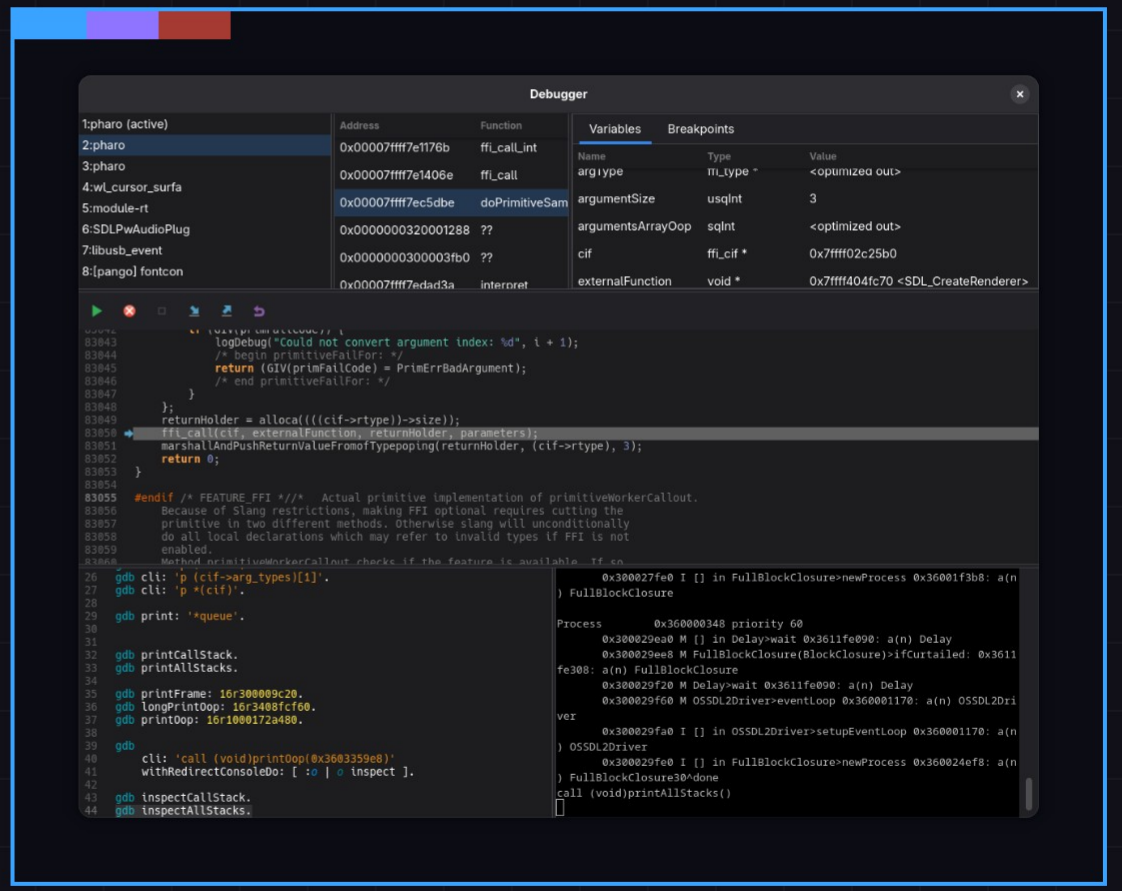
Pdbg

THE STAKES RISE

With FFI-heavy applications, some failures do not become Pharo exceptions.

They become process crashes.

At that point, Pharo debugging becomes **GDB debugging**.



Applications are integration tests for Pharo

Frameworks become real in use

Widgets, editors, lists, terminals, perspectives.

Boundaries become visible

Packaging, dependencies, native APIs, threads, process failures.

Abstractions become sharper

Requirements discovered by use are harder to ignore.

Note: Packaging is still unfinished

But we ill arrive there!

This talk is not mainly about packaging.

But every application came back to it:

- smaller images
- dependency closure
- native libraries
- distribution
- cross-platform expectations

The missing tool is still clear:

open → configure → package

A small, practical path from image to desktop application.

Build the applications that make the imperfections impossible to ignore.

Every tool began as a personal annoyance. Every annoyance revealed a missing capability. Every missing capability became a chance to make Pharo better.