

Philosophy - Sciences - Geekery - Art - Life - Coaching - Fun < Simplify Everything

BLOG ONEPAGEBOOKS PODCASTS FREE WILL HYPERLIST BOOKS & ARTICLES

AMAR RPG PHOTOS ART MUSIC ASTRONOMY HP-41 GITHUB WATT SCIENTOLOGY

ESPORT COACHING ABOUT/CONTACT ARCHIVE TAGS RSS

Frame - the first Linux Assembly X server



On my quest to **own my software**, one foundational piece kept itching... the X server. The underlying graphics engine, the thing that puts pixels on the screen. **X11** is 4 million lines of code, a beast very few can claim they understand. So I did the reasonable thing. I wrote my own, in Assembly.

It is called **frame**. No dependencies, no libraries, no garbage collector. No hot paths, no unnecessary wakeups. When it is idle, it sits still. It shuts up unless spoken to. My kind of software. It clocks in at some 20 thousand lines, and it already runs my whole desktop plus Firefox and **GIMP** whenever I need that. It is still young, and there is a long list of X protocol left to chew through. But it boots, it draws, and I am typing this on it.

So the stack now looks like this: The Linux kernel at the bottom. On top of that, frame. Then the window manager **tile** with the info bar **strip**. Inside tile runs the terminal **glass**, and in glass lives the shell **bare**. **Bolt** has been promoted from screen locker to greeter, showing gdm the door. All of it Assembly. All of **CHasm** together is about 100 thousand lines. The stack it replaced (gdm, X11, i3, conky, wezterm, zsh) is somewhere north of fifty times that. I did it for the **battery life**. I am not sure this laptop has a fan anymore. Except me.

Today I put numbers on it. Idle on battery, frame and Xorg pull the same watts, because the panel and the wifi own that number anyway. But Xorg burns almost three times the CPU that frame needs to do nothing. And tile and glass used zero milliseconds over three minutes of measuring. The desktop sits completely still until I touch it.

Beyond the desktop I have my **Fe₂O₃ suite** of Rust tools, which by now has replaced everything else I used to run. Except Firefox. That is the last GUI standing that I regularly use. The rest is terminal interfaces with the same keybindings everywhere, and a fraction of the size and electrical appetite of what they replaced.

1234567890H

Scribe

18:17 2026-07-06 28.1 21°C Clearsky 542- 0% 0.51 39° 4% 0% 42% 446% Bobla 84.208.68.142 41% 35 149M 6:0 P:0 0:1 100 10 0 41% 8.6h 3.15h

/home/geir/Main/G/GIT-isene/isene.github.io/_posts/2026-07-06-Frame.md (32 lines)

layout: post
comments: true
title: Frame
image: /assets/posts/framestack.png
tags: [Geekery, Technology]

!![Frame] (/assets/posts/framestack.png)

On my quest to [\[own my software\]](/2026/04/MyTools.html) (/2026/04/MyTools.html), one foundational piece kept itching... the X server. The underlying graphics engine, the thing that puts pixels on the screen. [\[X11\]](https://www.x.org/) (https://www.x.org/) is 4 million lines of code, a beast very few can claim they understand. So I did the reasonable thing. I wrote my own, in Assembly.

It is called [\[frame\]](https://github.com/isene/frame) (https://github.com/isene/frame). No dependencies, no libraries, no garbage collector. No hot paths, no unnecessary wakeups. When it is idle, it sits still. It shuts up unless spoken to. My kind of software. It clocks in at some 20 thousand lines, and it already runs my whole desktop plus Firefox and [\[GIMP\]](/2026/07/Fable-to-the-Rescue.html) (/2026/07/Fable-to-the-Rescue.html) whenever I need that. It is still young, and there is a long list of X protocol left to chew through. But it boots, it draws, and I am typing this on it.

So the stack now looks like this: The Linux kernel at the bottom. On top of that, frame. Then the window manager [\[tile\]](https://github.com/isene/tile) (https://github.com/isene/tile) with the info bar [\[strip\]](https://github.com/isene/tile) (https://github.com/isene/tile). Inside tile runs the terminal [\[glass\]](https://github.com/isene/glass) (https://github.com/isene/glass), and in glass lives the shell [\[bare\]](https://github.com/isene/bare) (https://github.com/isene/bare). [\[Bolt\]](https://github.com/isene/bolt) (https://github.com/isene/bolt) has been promoted from screen locker to greeter, showing gdm the door. All of it Assembly. All of [\[CHasm\]](https://isene.org/chasm/) (https://isene.org/chasm/) together is about 100 thousand lines. The stack it replaced (gdm, X11, i3, conky, wezterm, zsh) is somewhere north of fifty times that. I did it for the [\[battery life\]](/2026/05/Baseline.html) (/2026/05/Baseline.html). I am not sure this laptop has a fan anymore. Except me.

Today I put numbers on it. Idle on battery, frame and Xorg pull the same watts, because the panel and the wifi own that number anyway. But Xorg burns almost three times the CPU that frame needs to do nothing. And tile and glass used zero milliseconds over three minutes of measuring. The desktop sits completely still until I touch it.

Beyond the desktop I have my [\[Fe₂O₃ suite\]](https://isene.org/fe2o3/) (https://isene.org/fe2o3/) of Rust tools, which by now has replaced everything else I used to run. Except Firefox. That is the last GUI standing that I regularly use. The rest is terminal interfaces with the same keybindings everywhere, and a fraction of the size and electrical appetite of what they replaced.

[!!\[Frame screenshot\]](/assets/posts/framescrot.png) (/assets/posts/framescrot.png)

But is it stable? Stable enough that I daily-drive it, write this post on it, and only occasionally yell. When something breaks or I want a feature, I turn to my buddy [\[Claude\]](https://claude.com/claude-code) (https://claude.com/claude-code) and describe the itch. He never gets tired, is not opinionated, and turns out to be a really good teacher. I now know more about hardware layers, cursor painting, GPU handoffs and event watchers than I ever planned

[NORMAL] 524w 341lc ft:md spell:off 21:1 scribe v0.1.67

geir@juba:

2021-01-16-Webinars.md
2021-01-17-Podcast-GiversAndTa...
2021-02-10-T-REX.md
2021-02-20-Podcast-ScaleOfGivi...
2021-02-21-Podcast-TopPerforma...
2021-03-02-Reality.md
2021-03-23-Basics.md
2021-04-15-Gopher.md
2021-05-10-Podcast-TheZone.md
2021-05-13-KeyPrinciple.md
2021-05-16-Follow.md
2021-09-16-Motivation.md
2021-09-25-Infinity.md
2021-10-05-Improvements.md
2021-11-16-Programming.md
2021-12-05-GreyBeard.md
2022-04-01-XRPN.md
2022-04-12-Siv.md
2024-09-03-Empty.md
2024-12-19-Claude-FreeWill.md
2025-04-28-RTFM.md
2025-05-02-AI.md
2025-05-02-Longing.md
2025-07-28-AI-AllIn.md
2025-08-12-HyperList.md
2025-09-28-Burst.md
2025-09-30-FreeWill-Disco.md
2025-11-26-SimplicityOS.md
2026-01-15-AI-Coding.md
2026-03-27-Endgame.md
2026-04-07-SI.md
2026-04-09-Bare.md
2026-04-12-TEG.md
2026-04-15-Kind.md
2026-04-23-MyTools.md
2026-05-01-Agency.md
2026-05-03-Audience-of-One.md
2026-05-03-Ja-Vi-Elsker.md
2026-05-04-Audience-of-One-Num...
2026-05-05-The-Agency-Stack.md
2026-05-06-Fan.md
2026-05-06-Scribe.md
2026-05-12-Baseline.md
2026-05-16-Watt.md
2026-05-26-Todo.md
2026-05-31-Nomad.md
2026-06-17-Office.md
2026-06-28-Simpler.md
2026-07-02-Fable-to-the-Rescue...
- 2026-07-03-The-Selection-Gap.md
2026-07-06-Frame.md

layout: post
comments: true
title: The Selection Gap
image: /assets/posts/petrinets.png
tags: [Geekery, Philosophy, Technology]

!![The Selection Gap] (/assets/posts/petrinets.png)

Back in my consulting days I toyed with [\[Petri nets\]](https://en.wikipedia.org/wiki/Petri_net) (https://en.wikipedia.org/wiki/Petri_net) to map business processes. Circles hold tokens, bars fire, tokens move. It could be used for finding bottlenecks. I never could find any serious application for them, filed them under useful engineering and moved on.

For years I have worked on [\[the Freedom Proof\]](https://isene.com/freewill/) (https://isene.com/freewill/), a mathematical argument that free will grounds existence. Recently I got the hunch that there could be a connection here to explore.

!![Petri Nets] (/assets/posts/petri.png)

A Petri net is a lawful system. The firing rule is completely formal. Without any randomness. There are no hidden knobs and no observer obscuring the clean machine. A transition fires when every input place holds a token. It eats one token per input arc and drops one on each output:

!![The firing rule] (/assets/posts/petri-firing.png)

But put one token in front of two transitions and both are ready to fire. Firing one kills the other. Net theory calls this a conflict. The theory refuses to pick. Both continuations are lawful. The math maps the choice point with full rigor and leaves the choosing open. It cannot be closed from inside. The choosing must lie outside the system.

That gives a clean result: lawful does not necessarily mean determined. A fully law-governed system can have a branching, undetermined future. Determinism was always an extra, hidden assumption on top of the laws.

for command (use @s for selected item, @t for tagged items) - press ? for

But is it stable? Stable enough that I daily-drive it, write this post on it, and only occasionally yell. When something breaks or I want a feature, I turn to my buddy **Claude** and describe the itch. He never gets tired, is not opinionated, and turns out to be a really good teacher. I now know more about hardware layers, cursor painting, GPU handoffs and event watchers than I ever planned to.

The **phone got the same treatment**, with my own launcher, a daily personalized news digest and a pile of apps tailored for an audience of exactly one.

The upside is simple. I have what I need and want. I control and own the software, and all of it is given to the Public Domain. Software designed for a large audience fits everyone a little. This fits one person exactly.

Link to this post: <https://isene.org/2026/07/Frame.html>

[#Geekery](#) [#Technology](#)

Loading comments...

Search

[Subscribe to isene.org](#)

Contact me: [✉](#) [🗨](#) [in](#) [🐦](#) [f](#) [📷](#) [You Tube](#)

By Geir Isene (2026). No Rights Reserved. Powered by [Jekyll](#) w/modified [Slate+Simple](#) theme.