The Foundry of Builder

Christian Hergert
July 2025

GNOME™

Builder (revisited)

- Started a little over 10 years ago to modernize GNOME app development
- The first IDE to abstract containers as a first class primitive
- Is a rather large GTK application (275k LOC + ~150K in broken off libraries)
- Has birthed many new projects out of the code-base
- Somewhat limiting because you need to be inside of Builder to take advantage
- Many people use Builder to create project templates then jump to their editors of choice (VS Code, Vim, Emacs, etc)
- Code from Builder has ended up all over the place in GNOME and GTK

GNOME™

Side Quests

- To create Builder I had to create a lot of other tooling
  - libdazzle, jsonrpc-glib, template-glib, libdex, manuals, deviced, codesearch, d-spy, gom, libspelling, libpanel (probably missing some others)
- In some cases I had to become maintainer to keep the projects alive which involved a great deal of new features and tech-debt reduction
  - gtksourceview, libpeas
- Some new core/devel apps along the way born of Builder code
  - gnome-text-editor, sysprof, ptyxis, manuals, d-spy
- Invented a new data-structure (piecetable/b+tree hybrid) while also using existing knowledge for others (fast-fuzzy search w/ O(n*m) worst case)
- Do multi-process the hard way before LSP/DAP w/ D-Bus and socketpair()

Side Quests (Cont...)

- Sometimes you have to work upstream
  - We need macOS support so write a new GDK backend w/ GPU support
  - The GPU renderer was increasingly difficult to improve, so rewrite it with all the knowledge acquired in its initial creation (since done again by B. Otte)
  - We needed faster terminal rendering so add GPU support to VTE
  - Profiling was terrible on x86_64 so lobby distros for frame-pointers
- Completely redesign how we do async and IO in C (libdex) including futures/fibers, poll-able semaphores, await, work-stealing threadpools, io_uring and more while supporting Linux, macOS, TheBSD(s), Solaris/Illumos, and Windows

Side Quests (Cont…)

- Some side-quests are still in progress today for future features
  - Libmks, drafting, (and a few more TBA)
- Wrote a book on how Builder works so others can contribute

GNOME™

I’m Exhausted

GNOME™

Why

- Often times GNOME bureaucratic processes exhaust any energy I had left
- Having applications depend on other applications is a non-goal of many distributions, nor does it play well with sand-boxing
- Keeping up with releases is Actually a Lot of Work
- More than half of issues filed are extremely low quality due to lacking details, system information, language barriers, aggressive language, condescending attitudes, or just flat out entitlement

GNOME™

I can fix some of that

GNOME™

Foundry

- A new CLI, static, or shared library. Basically an IDE in a box including plug-ins
- All of the core abstractions from Builder, built on libdex and makes heavy use of futures, fibers, and thread-pools
- Feature flags to compile in/out what you want for static library consumption
- CLI tooling lets us unit test IDE features without starting up a GTK app
- Allows for sharing a lot of Builder code across current/future applications
  - Builder, Drafting, Manuals, Sysprof initially, some new ones TBA
- Service/Manager oriented, even more so than Builder
  - Services for Builds, Docs, VCS, Files, Text, Diagnostics, Dependencies, LSPs, DAPs, Devices, SDKs, Reflection, Settings, Tests, Debuggers/Profilers, Search, Commands, App runners, CLI tooling, subprocess IPC/coordination, LLM/MCP possible

Foundry CLI

- CLI can even be used like an IDE using “foundry enter”
- Spawns a sub-shell on your PTY beneath the foundry command
- Running Foundry commands in sub-shell proxies command to the parent Foundry process allowing long running background jobs and low-latency
- You do not need to use “foundry enter”, in other cases it will just incur a very small startup time cost (generally < 100msec) for subsystems to settle such as device/SDK managers
- Lots of tooling is already ready, in fact I do most development with Foundry now when I’m in a terminal

GNOME™

Foundry CLI (Cont…)

- Try it now on a project that you use in Builder

```
$ foundry clone git@gitlab.gnome.org:chergert/ptyxis
$ cd ptyxis/
$ foundry run
```

GNOME™

Foundry-GTK

- A new static or shared library that bridges Foundry APIs to GTK
- Also links with some most-likely used libraries for easy integration
  - GtkSourceView for text editing
  - VTE for terminals w/ palette support similar to Ptyxis
  - Libspelling
  - WebKit
- Does not link against libadwaita to allow for use by non-GNOME, but GTK-consuming platforms (Elementary, XFCE, MATE, Cinnamon, etc). This provides the inner workings, not your UI/HIG/Patterns

Come and Help!

GNOME™

Lots to Work On

- A lot of core services are in place which make it more obvious how things work
- Come take a look at it and start contributing
- If you tinker with it please help me write documentation/FAQs/etc
- The new developer app Manuals is a great example if you want to see how to consume it from an application
- Builder to get re-based on Foundry and fully expect to cut down the code-size by more than half (which was already halved once in GTK 3 → 4 transition)
- Write specialized tools using specific subsystems (debugger UI stands out as fairly obvious) or new plug-ins
- Find ways to use LLM/MCP with ethically sourced local models to automate the more tedious parts of software creation (summaries for releases? does my issue comment sound rude? Anything?)

Thank You!

https://gitlab.gnome.org/GNOME/foundry/