Gnome Planet - Latest News

- Andy Wingo: on the impossibility of composing finalizers and ffi (2024/02/26 10:05)
  While poking the other day at making a Guile binding for Harfbuzz, I remembered why I don’t much do this any more: it is impossible to compose GC with explicit ownership.Allow me to illustrate with an example. Harfbuzz has a concept of blobs, which are refcounted sequences of bytes. It uses these in a number of places, for example when loading OpenType fonts. You can get a peek at the blob’s contents back with hb_blob_get_data, which gives you a pointer and a length.Say you are in LuaJIT. (To think that for a couple years, I wrote LuaJIT all day long; now I can hardly remember.) You get a blob from somewhere and want to get its data. You define a wrapper for hb_blob_get_data:local hb = ffi.load("harfbuzz") ffi.cdef [[typedef struct hb_blob_t hb_blob_t; const char * hb_blob_get_data (hb_blob_t *blob, unsigned int *length);]] Presumably you then arrange to release LuaJIT’s reference on the blob when GC collects a Lua wrapper for a blob:ffi.cdef [[void hb_blob_destroy (hb_blob_t *blob);]] function adopt_blob(ptr) return ffi.gc(ptr, hb.hb_blob_destroy) end OK, so let’s say we get a blob from somewhere, and want to copy out its contents as a byte string.function blob_contents(blob) local len_out = ffi.new('unsigned int') local contents = hb.hb_blob_get_data(blob, len_out) local len = len_out[0]; return ffi.string(contents, len) end The thing is, this code is as correct as you can get it, but it’s not correct enough. In between the call to hb_blob_get_data and, well, anything else, GC could run, and if blob is not used in the future of the program execution (the continuation), then it could be collected, causing the hb_blob_destroy finalizer to release the last reference on the blob, freeing contents: we would then be accessing invalid memory.Among GC implementors, it is a truth universally acknowledged that a program containing finalizers must be in want of a segfault. The semantics of LuaJIT do not prescribe when GC can happen and what values will be live, so the GC and the compiler are not constrained to extend the liveness of blob to, say, the entirety of its lexical scope. It is perfectly valid to collect blob after its last use, and so at some point a GC will evolve to do just that.I chose LuaJIT not to pick on it, but rather because its FFI is very straightforward. All other languages with GC that I am aware of have this same issue. There are but two work-arounds, and neither are satisfactory: either develop a deep and correct knowledge of what the compiler and run-time will do for a given piece of code, and then pray that knowledge does not go out of date, or attempt to manually extend the lifetime of a finalizable object, and then pray the compiler and GC don’t learn new tricks to invalidate your trick.This latter strategy takes the form of “remember-this” procedures that are designed to outsmart the compiler. They have mostly worked for the last few decades, but I wouldn’t bet on them in the future.Another way to look at the problem is that once you have a system working—though, how would you know it’s correct?—then you either never update the compiler and run-time, or you become fast friends with whoever maintains your GC, and probably your compiler too.For more on this topic, as always Hans Boehm has the first and last word; see for example the 2002 Destructors, finalizers, and synchronization. These considerations don’t really apply to destructors, which are used in languages with ownership and generally run synchronously.Happy hacking, and be safe out there!
- Flathub Blog: Introducing App Brand Colors (2024/02/26 00:00)
  We're gearing up to launch curated banners on the Flathub home page! However, before we can do that there's one more blocker: Banners need a background color for each app, and many apps don't provide this metadata yet. This is why today we're expanding our MetaInfo quality guidelines and quality checks on the website; If you haven't yet, please add these colors to your app's MetaInfo file using the <branding/>

appstream tag, and read on to learn more about brand colors. What are brand colors? App brand colors are an easy and effective way for app developers to give their listing a bit more personality in app stores. In combination with the app icon and name, they allow setting a tone for the app without requiring a lot of extra work, unlike e.g. creating and maintaining additional image assets. Why now? This idea was first implemented in elementary AppCenter, and later standardized as part of the AppStream specification. While it has been in AppStream itself for a few years, it was unfortunately not possible for Flathub's backend to pick it up until the recent port to libappstream. This is why many apps are still not providing this metadata—even if it was available from the app side we were unable to display it until now. Now that we can finally pick these colors up from AppStream MetaInfo files, we want to make use of them—and they are essential for the new banners. Adding brand colors Apps are expected to provide two different brand colors for light and dark. Here's an example of a MetaInfo file in the wild including brand colors. This is the snippet you need to include in your MetaInfo file: <branding> <color type="primary" scheme_preference="light">#faa298</color> <color type="primary" scheme_preference="dark">#7f2c22</color></branding> In choosing the colors, try to make sure the colors work well in their respective context (e.g. don't use a light yellow for the dark color scheme), and look good as a background behind the app icon (e.g. avoid using exactly the same color to maintain contrast). In most cases it's recommended to pick a lighter tint of a main color from the icon for the light color scheme, and a darker shade for the dark color scheme. Alternatively you can also go with a complementary color that goes well with the icon's colors. What's next? Today we've updated the MetaInfo quality guidelines with a new section on app brand colors. Going forward, brand colors will be required as part of the MetaInfo quality review. If you have an app on Flathub, check out the guidelines and update your MetaInfo with brand colors as soon as possible. This will help your app look as good as possible, and will make it eligible to be featured when the new banners ship. Let's make Flathub a more colorful, exciting place to find new apps!

- Jussi Pakkanen: Creating tagged PDFs with CapyPDF now sort of possible (2024/02/23 19:50)
  There are many open source PDF generators available. Unfortunately they all have some limitations when it comes to generating tagged PDFsCairo does not support tagged PDFs at allLaTeX can create tagged PDFs, but obviously only out of LaTeX documentsScribus does not support tagged PDFLibreOffice does support tagged PDF generation, but its code is not available as a standalone library, it can only be used via LibreOfficeHexaPDF does not support tagged PDFs (though they seem to be on the roadmap), but it is implemented in Ruby so most projects can't use itFirefox can print pages to PDF, but the result is not tagged, even though the PDF document structure model is almost exactly the same as in HTMLThere does not seem to be a library that provides for all of this with a "plain C" API that can be used to easily generated tagged PDFs using almost any programming language.There still isn't, but at least now CapyPDF can generate simple tagged PDF documents. A sample document can be downloaded via this link. Here is a picture showing the document structure in Acrobat Pro.It should also work in things like screen readers and other accessibility tools, but I have not tested it.None of this is exposed in the C API, because this has a fairly large API surface and I have not yet come up with a good way to represent it.
- Felix Häcker: #136 New Papers (2024/02/23 00:00)
  Update on what happened across the GNOME project in the week from February 16 to February 23. Sovereign Tech Fund Sonny says As part of the GNOME STF (Sovereign Tech Fund) project, a number of community members are working on infrastructure related projects. Accessibility Joanie continued improving the Orca screen reader: Finished removing all pyatspi code □ New Keyhandling/Grabs Event Manager: Add means to pause and clear the event queue: https://gitlab.gnome.org/GNOME/orca/-/commit/967a32407 Check for device before adding a grab: https://gitlab.gnome.org/GNOME/orca/-/commit/c4a7b391f Remove/Move speech-related hacks See related

https://gitlab.gnome.org/GNOME/orca/-/merge_requests/182#note_2009083 speech-dispatcher: Remove hack for newline followed by period: https://gitlab.gnome.org/GNOME/orca/-/commit/3a4001df6 speech-dispatcher: Move u00a0-> to adjustForPronunciation: https://gitlab.gnome.org/GNOME/orca/-/commit/1957edb8a Event spam + code clean up Discovered and filed https://gitlab.gnome.org/GNOME/gtk/-/issues/6449 Took the opportunity to clean up the Orca code that handles such issues: https://gitlab.gnome.org/GNOME/orca/-/commit/8faea2f7f Ignore checked-changed events from non-showing widgets: https://gitlab.gnome.org/GNOME/orca/-/commit/48ebd3db2 Georges changed how WebKit makes accessible objects implement Hypertext, Hyperlink, and Text. This should unblock further accessibility work for Joanie and Orca: https://github.com/WebKit/WebKit/pull/24956 Andy landed Spiel text-to-speech support in Orca: Update Orca’s Spiel server along with change to upstream API: SpielProvider changes: https://gitlab.gnome.org/GNOME/orca/-/merge_requests/182/diffs?commit_id=4db5f96178652298b563c9e829689952cdc0fb4b New events: https://gitlab.gnome.org/GNOME/orca/-/merge_requests/182/diffs?commit_id=b1e66999690d88bb4cde5093ec6d958472cc80d8 Build Spiel from upstream: https://gitlab.gnome.org/GNOME/orca/-/commit/a93d512c3f37511c90c8aab68c14e1c8e526b722 Security Dhanuka continued work on implementing oo7-daemon: Opened a new PR to faciliate collaboration with Bilal https://github.com/bilelmoussaoui/oo7/pull/73 Make Algorithm enum public under unstable feature: https://github.com/bilelmoussaoui/oo7/pull/72 Removed OnceLock<Keyring> and introduced Service::new() to initialize the Service containing oo7::portal::Keyring Ported to zbus 4.0 New Accessibility Stack (Newton) Matt is working on the AT-SPI compatibility library for Newton. Extracted AccessKit AT-SPI implementation into a core library https://github.com/AccessKit/accesskit/pull/352 as a prerequisite Hardware Support Jonas continued his work on various GNOME Shell things that are still on track for 46, including hardware encoding for screencasts and the new gesture API: Debugged some more gnome-shell screencast issues with HW encoding and iterated on the MR (https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/2080), it will hopefully land once the gstreamer pipelines get an ack from the gstreamer folks Rebased gestures part 2 MR: https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/2389 Helped the Outreachy students with mobile QA Jonas continued to push fractional scaling forward: Almost finished (only needs more tests) with the mutter monitor config work to allow toggling on “scale-monitor-framebuffer” by default: https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3596 Sketched out two algorithms for migrating monitor configs in mutter and implemented them (see https://gitlab.gnome.org/GNOME/mutter/-/commit/31c3d647d9ac792cc3b07f653dd4dcba75b8fc32?merge_request_iid=3596 for fancy ASCII art) Updated Jonas Ådahl’s MR to g-s-d for Xwayland scaling: https://gitlab.gnome.org/GNOME/gnome-settings-daemon/-/merge_requests/353 Opened an MR to g-c-c for to configure Xwayland scaling: https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2286 Worked with Tobias and Sonny on design for Xwayland scaling option in g-c-c Dor continued his work on variable refresh rate support: Made Mutter’s libdisplay-info dependency more visible to downstream packagers Among other features, this is useful for https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3576 Mutter MR: https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3582 g-b-m MR: https://gitlab.gnome.org/GNOME/gnome-build-meta/-/merge_requests/2671 Rebased MRs on GNOME 46 beta and made minor changes following review comments: https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/1154 https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/734 https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3576 https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2260 Georges is investigating Nvidia GPU issues with WebKitGTK Ran a variety of WebKitGTK tests, builds, and apps on Nvidia Reduced the scope of the bugs as a lot more seems to be functional now. GTK3 and GStreamer videos might still have problems, working on it https://bugs.webkit.org/show_bug.cgi?id=228268 https://bugs.webkit.org/show_bug.cgi?id=261874 :information_source: we are trying to make things work but this is not an endorsement that you should use Nvidia on Linux :) Platform Alice fixed some minor issues before the libadwaita 1.5 release Fixed dialog dimming; fixed shade colors in dark in general in process: https://gitlab.gnome.org/GNOME/libadwaita/-/merge_requests/1052 Fixed .devel styles on dialogs: https://gitlab.gnome.org/GNOME/libadwaita/-/merge_requests/1051 Evan worked on merging the two GNOME TypeScript bindings https://github.com/gjsify/ts-for-gir/pull/144 Evan started prototyping necessary changes in meson to move GNOME repos over to the GLib-based GI compiler Andy put some finishing touches on the GNOME Online Accounts GTK4 port: Design discussion: https://gitlab.gnome.org/Teams/Design/settings-mockups/-/issues/68 Resulting MRs: https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/178, https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2284 Julian continued his work on improving notifications: Worked on refactoring notification code in GNOME Shell and expanding notification in calendar drawer https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/3173 Looked into notification startup id/activation token: Use correct platform data when activating application actions https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/3198 gappinfo: Pass activation token from launch context to open_uri/file portal https://gitlab.gnome.org/GNOME/glib/-/merge_requests/3933 Hubert is working on the device permission backward compatibility https://github.com/flatpak/flatpak/issues/5681 Hubert landed memory leak fixes https://github.com/flatpak/flatpak/pull/5683 Home Encryption Adrian made major progress on integrating systemd homed for home encryption: Landed blob directories in systemd ▯ https://github.com/systemd/systemd/pull/30840 Fixed GDM hang when trying to unlock user locked & frozen by homed https://gitlab.gnome.org/GNOME/gdm/-/merge_requests/235 Fixed blob-dir-related systemd bug: https://github.com/systemd/systemd/issues/31417 https://github.com/systemd/systemd/pull/31419 Contributed to design discussion about passwordless users https://gitlab.gnome.org/Teams/Design/settings-mockups/-/issues/66 Opened an RFE about the lack of support for auditing in systemd-homed https://github.com/systemd/systemd/issues/31447 Brought homed up to feature parity with legacy users in gnome-control-center https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2306 Published a GNOME OS branch for testing and ironed out some isues https://gitlab.gnome.org/GNOME/gnome-build-meta/-/merge_requests/2681) https://gitlab.com/freedesktop-sdk/infrastructure/freedesktop-sdk-docker-images/-/merge_requests/463 https://gitlab.gnome.org/GNOME/gnome-build-meta/-/merge_requests/2661 Tracked down a systemd regression that completely breaks GNOME https://github.com/systemd/systemd/issues/31287 Continue work and addressed review on Homed update policy v2 https://github.com/systemd/systemd/pull/31153 Landed new user record language fields, so that we can represent users who can speak multiple languages https://github.com/systemd/systemd/pull/31206 Made gnome-initial-setup support homed https://gitlab.gnome.org/GNOME/gnome-initial-setup/-/merge_requests/239) Continued LKML discussion about deleting data out of memory on homed lock https://lore.kernel.org/r/20240116-tagelang-zugnummer-349edd1b5792@brauner Rebased & finished up PR that makes systemd freeze user sessions more aggressively https://github.com/systemd/systemd/pull/30612 GNOME Incubating Apps Sophie (she/her) reports Papers has been accepted into the GNOME Incubator. The GNOME incubation process is for apps that are designated to be accepted into GNOME Core or GNOME Development Tools if they reach the required maturity. Papers is a fork of GNOME’s current document viewer Evince. Due to limited resources, the Evince project can currently not facilitate larger changes like the port to GTK 4 and Libadwaita. Those goals will now be pursued as part of the Papers project. Papers has already been ported to GTK 4 and Libadwaita. The incubation progress will be tracked in this issue. GNOME

Circle Apps and Libraries Gaphor A simple UML and SysML modeling tool. Arjan says Last week Dan Yeaw release Gaphor 2.24.0, the user friendly SysML/UML modeling application. Highlights of this release are: Gaphor is now REUSE compliant. This makes it easier for third-parties to build on top of Gaphor. Improvements in the CSS rendering. Styling can be applied much more fine grained. Ear Tag Edit audio file tags. knuxify reports Ear Tag 0.6.0 has been released! Among other changes. this release brings: various improvements to the "Rename Selected Files" feature, such as syntax highlighting and the ability to move renamed files to a folder (and create subfolders from tags); a new "Extract Tags from Filename" option that can automatically extract tags based on a pattern; options to remove all tags from a file or undo all currently pending changes; …and a few small design tweaks. Read more about this release on the 0.6.0 release page, or get the latest version from Flathub. Third Party Projects Maximiliano □ reports search-provider, your favorite Rust zbus-based library to interact with GNOME's SearchProvider, just released its version 0.8.1, new in this version we add a new feature to turn gdk::Textures into icons that you can send over the bus, no more meddling with gdk_pixbuf::Pixbufs! slomo reports The GStreamer team is excited to announce the first release candidate for the upcoming stable 1.24.0 feature release. This 1.23.90 pre-release is for testing and development purposes in the lead-up to the stable 1.24 series which is now frozen for commits and scheduled for release very soon. Depending on how things go there might be more release candidates in the next couple of days, but in any case we're aiming to get 1.24.0 out as soon as possible. Preliminary release notes highlighting all the new features, bugfixes, performance optimizations and other important changes will be available in the next few days. If you notice any problems, please file an issue in GitLab. https://discourse.gstreamer.org/t/gstreamer-1-23-90-pre-release-1-24-0-rc1 Thanks for testing! ghani1990 announces This week, Alain unveiled Planify 4.5, bringing a wave of exciting design enhancements and powerful new features to the popular task management app. Boost your productivity with these innovative additions: Seamless Nextcloud Integration. Simplified Task Migration from Planner Enhanced Task Management: Experience the convenience of drag-and-drop task movement, allowing you to effortlessly organize your workload with a simple click and drag. Clearer Visibility: Gain valuable insights with the "Always Show Completed Subtasks" option, keeping track of your progress and ensuring nothing falls through the cracks. Efficient Task Creation: Introducing the "Create more" feature, enabling you to swiftly add multiple tasks without interrupting your workflow. Experience a Smoother User Journey: Planify 4.5 doesn't stop there. It boasts a sleekly redesigned date and time selection widget, further enhancing user experience. Additionally, several bug fixes have been implemented to ensure optimal performance and a seamless workflow. GNOME Websites federico reports GNOME's Code of Conduct has moved from the wiki to https://conduct.gnome.org. All the informational pages about the CoC Committee, procedures, etc. have moved from the wiki to https://conduct.gnome.org/committee Shell Extensions Just Perfection reports The extensions port guide for GNOME Shell 46 is ready. If you need any help with your extension, please ask us on GNOME Extensions Matrix Channel. Miscellaneous barthalion announces Flathub's automatic build validation is more thorough now, and includes checks for issues we previously would have only flagged manually. We have also started moderating all permission changes and some critical MetaInfo changes. Last but not least, we have switched to libappstream, enabling specifying supported screen sizes for mobile devices, and other features available in the latest spec. More details on our blog: https://docs.flathub.org/blog/improved-build-validation/ Bilal Elmoussaoui says Following the release of zbus 4.0, I have released a new version of ashpd and oo7. The releases consist of mostly bug fixes and reduced dependencies thanks to the zbus update. On top of that, oo7-cli, a secret-tool replacement is now available to install with cargo install oo7-cli That's all for this week! See you next week, and be sure to stop by #thisweek:gnome.org with updates on your own projects!

- Dorothy Kabarozi: Conversations in Open Source: Insights from Informal chats with Open Source contributors. (2024/02/21 05:06)

Introduction Open source embodies collaboration, innovation, and accessibility within the technological realm. Seeking personal insights behind the collaborative efforts, I engaged in conversations with individuals integral to the open source community, revealing the diversity, challenges, and impacts of their work. Conversations Summary Venturing beyond my comfort zone, I connected with seasoned open source contributors, each offering unique perspectives and experiences. Their roles varied from project maintainers to mentors, working on everything from essential libraries to innovative technologies. Das: Shared valuable insights on securing roles in open source, including resources for applications and tips for academic writing and conference speaking. The best part with Das was that she also reviewed my resume and shared many ways i could make it outstanding and shared templates to use for this.We had a really great chat. Samuel: A seasoned C/C++ programmer working mainly on open-source user-space driver development.He was kind enough to share his 20 years long journey on how he started working with open source and how he loves working with low level Hardware.He also commended Outreachy as a great opportunity and my contributions with GNOME in QA testing.He encouraged me to apply for roles in the company he's working with,and highlighted "Even if they say NO now, next time they will say YES". Samuel also encouraged me to find my passion and this will guide me to learn faster,create my personal brand and encouraged me to submit some conference talks. Dustin: Shared his 20 year journey and we mostly talked about Programming and software engineering in general . Highlighted the significance of networking and adaptability to learn quickly in open source.He shared a story how he "printed out code on one of his first jobs and learnt a skill of figuring out early what you don't need to understand when faced with a big code base". This is one skill I needed at the start instead of drowning in documentation trying to understand the project and where to start. Stefan: Discussed his transition from a GSOC participant to a mentor,shared opensource job links , commended Outreachy as a big plus. He highlighted not to set yourself up by mental blocking that you can't do anything, because you can.He encouraged to submit talks at conferences, network and publishing my work. These interactions showcased the wide-ranging backgrounds and motivations within the open source community and have deepened my respect for the open source community and its contributors. I have some homework to do with my resume and the links to opportunities that were shared with me.Open source welcomes contributors at all levels, offering a platform for innovation and collective achievement. Feel free to be an Outreachy intern on the upcoming cohorts to start your journey. Best of luck.

- Flathub Blog: Improved build validation, increased moderation, and the long-awaited switch to libappstream (2024/02/21 00:00)
  Flathub's automatic build validation is more thorough now, and includes checks for issues we previously would have only flagged manually. There is a chance that if your app has been passing the continuous integration checks previously, it will fail now; here's why, and what to do about it. If your application no longer passes the build validation stage in either Buildbot (for apps maintained on GitHub) or flat-manager (for direct uploads), make sure to look for specific messages in the log. Explanations for various error messages can be found in the documentation. If you are interested in running the linter locally or integrating it with your own CI, please refer to the project page. We have also started moderating all permission changes and some critical MetaInfo changes. For example, if a build adds or removes a static permission (as seen in the finish-args array in the manifest) or changes the app's user-facing name, it will be withheld for manual human review. Reviewers may reject a build and reach out for clarification about the change. Flathub has also switched to a modern, well-maintained AppStream library, known as libappstream. This enables developers to use all features described in the AppStream 1.0 specification, including specifying supported screen sizes for mobile devices, or video snippets to accompany static screenshots. It also improves the validation of AppStream metadata. Many thanks to Philip Withnall, Luna Dragon and Hubert Figuière for their work on this across the Flatpak stack, and Matthias Klumpp for implementing knobs needed by Flathub in the library itself. This work has been ongoing since 2021. At one point along the way we briefly switched over to libappstream, but

had to revert due to unexpected breakage; however, today we are finally ready with all blocking issues addressed! While we were focused on closing the gaps to prevent potentially broken builds from being published, we regret that we failed to provide a heads-up about the coming validation changes. Any future breaking changes will be properly announced on this blog, and going forward we will also inform maintainers of affected apps about required changes in advance.

- Carlos Garcia Campos: A Clarification About WebKit Switching to Skia (2024/02/20 18:11)
  In the previous post I talked about the plans of the WebKit ports currently using Cairo to switch to Skia for 2D rendering. Apple ports don't use Cairo, so they won't be switching to Skia. I understand the post title was confusing, I'm sorry about that. The original post has been updated for clarity.
- Matthew Garrett: Debugging an odd inability to stream video (2024/02/19 22:30)
  We have a cabin out in the forest, and when I say "out in the forest" I mean "in a national forest subject to regulation by the US Forest Service" which means there's an extremely thick book describing the things we're allowed to do and (somewhat longer) not allowed to do. It's also down in the bottom of a valley surrounded by tall trees (the whole "forest" bit). There used to be AT&T copper but all that infrastructure burned down in a big fire back in 2021 and AT&T no longer supply new copper links, and Starlink isn't viable because of the whole "bottom of a valley surrounded by tall trees" thing along with regulations that prohibit us from putting up a big pole with a dish on top. Thankfully there's LTE towers nearby, so I'm simply using cellular data. Unfortunately my provider rate limits connections to video streaming services in order to push them down to roughly SD resolution. The easy workaround is just to VPN back to somewhere else, which in my case is just a Wireguard link back to San Francisco.This worked perfectly for most things, but some streaming services simply wouldn't work at all. Attempting to load the video would just spin forever. Running tcpdump at the local end of the VPN endpoint showed a connection being established, some packets being exchanged, and then… nothing. The remote service appeared to just stop sending packets. Tcpdumping the remote end of the VPN showed the same thing. It wasn't until I looked at the traffic on the VPN endpoint's external interface that things began to become clear.This probably needs some background. Most network infrastructure has a maximum allowable packet size, which is referred to as the Maximum Transmission Unit or MTU. For ethernet this defaults to 1500 bytes, and these days most links are able to handle packets of at least this size, so it's pretty typical to just assume that you'll be able to send a 1500 byte packet. But what's important to remember is that that doesn't mean you have 1500 bytes of packet payload - that 1500 bytes includes whatever protocol level headers are on there. For TCP/IP you're typically looking at spending around 40 bytes on the headers, leaving somewhere around 1460 bytes of usable payload. And if you're using a VPN, things get annoying. In this case the original packet becomes the payload of a new packet, which means it needs another set of TCP (or UDP) and IP headers, and probably also some VPN header. This still all needs to fit inside the MTU of the link the VPN packet is being sent over, so if the MTU of that is 1500, the effective MTU of the VPN interface has to be lower. For Wireguard, this works out to an effective MTU of 1420 bytes. That means simply sending a 1500 byte packet over a Wireguard (or any other VPN) link won't work - adding the additional headers gives you a total packet size of over 1500 bytes, and that won't fit into the underlying link's MTU of 1500.And yet, things work. But how? Faced with a packet that's too big to fit into a link, there are two choices - break the packet up into multiple smaller packets ("fragmentation") or tell whoever's sending the packet to send smaller packets. Fragmentation seems like the obvious answer, so I'd encourage you to read Valerie Aurora's article on how fragmentation is more complicated than you think. tl;dr - if you can avoid fragmentation then you're going to have a better life. You can explicitly indicate that you don't want your packets to be fragmented by setting the Don't Fragment bit in your IP header, and then when your packet hits a link where your packet exceeds

the link MTU it'll send back a packet telling the remote that it's too big, what the actual MTU is, and the remote will resend a smaller packet. This avoids all the hassle of handling fragments in exchange for the cost of a retransmit the first time the MTU is exceeded. It also typically works these days, which wasn't always the case - people had a nasty habit of dropping the ICMP packets telling the remote that the packet was too big, which broke everything.What I saw when I tcpdumped on the remote VPN endpoint's external interface was that the connection was getting established, and then a 1500 byte packet would arrive (this is kind of the behaviour you'd expect for video - the connection handshaking involves a bunch of relatively small packets, and then once you start sending the video stream itself you start sending packets that are as large as possible in order to minimise overhead). This 1500 byte packet wouldn't fit down the Wireguard link, so the endpoint sent back an ICMP packet to the remote telling it to send smaller packets. The remote should then have sent a new, smaller packet - instead, about a second after sending the first 1500 byte packet, it sent that same 1500 byte packet. This is consistent with it ignoring the ICMP notification and just behaving as if the packet had been dropped.All the services that were failing were failing in identical ways, and all were using Fastly as their CDN. I complained about this on social media and then somehow ended up in contact with the engineering team responsible for this sort of thing - I sent them a packet dump of the failure, they were able to reproduce it, and it got fixed. Hurray!(Between me identifying the problem and it getting fixed I was able to work around it. The TCP header includes a Maximum Segment Size (MSS) field, which indicates the maximum size of the payload for this connection. iptables allows you to rewrite this, so on the VPN endpoint I simply rewrote the MSS to be small enough that the packets would fit inside the Wireguard MTU. This isn't a complete fix since it's done at the TCP level rather than the IP level - so any large UDP packets would still end up breaking)I've no idea what the underlying issue was, and at the client end the failure was entirely opaque: the remote simply stopped sending me packets. The only reason I was able to debug this at all was because I controlled the other end of the VPN as well, and even then I wouldn't have been able to do anything about it other than being in the fortuitous situation of someone able to do something about it seeing my post. How many people go through their lives dealing with things just being broken and having no idea why, and how do we fix that?(Edit: thanks to this comment, it sounds like the underlying issue was a kernel bug that Fastly developed a fix for - under certain configurations, the kernel fails to associate the MTU update with the egress interface and so it continues sending overly large packets) comments

- Michael Meeks: 2024-02-19 Monday (2024/02/19 21:00)
  Calc stand-up, 1:1s with Miklos, Eloy, Andras. Marketing content call. Sync with Rash & Ash, chased patch merging, poked at a profile. Worked late, slept badly - a correlation I should avoid more.
- Carlos Garcia Campos: WebKitGTK and WPEWebKit Switching to Skia for 2D Graphics Rendering (2024/02/19 13:27)
  In recent years we have had an ongoing effort to improve graphics performance of the WebKit GTK and WPE ports. As a result of this we shipped features like threaded rendering, the DMA-BUF renderer, or proper vertical retrace synchronization (VSync). While these improvements have helped keep WebKit competitive, and even perform better than other engines in some scenarios, it has been clear for a while that we were reaching the limits of what can be achieved with a CPU based 2D renderer. There was an attempt at making Cairo support GPU rendering, which did not work particularly well due to the library being designed around stateful operation based upon the PostScript model—resulting in a convenient and familiar API, great output quality, but hard to retarget and with some particularly slow corner cases. Meanwhile, other web engines have moved more work to the GPU, including 2D rendering, where many operations are considerably faster. We checked all the available 2D rendering libraries we could find, but none of them met all our requirements, so we decided to try writing our own library. At the beginning it worked really well, with impressive results in performance even compared to other GPU based alternatives. However, it proved challenging to

find the right balance between performance and rendering quality, so we decided to try other alternatives before continuing with its development. Our next option had always been Skia. The main reason why we didn't choose Skia from the beginning was that it didn't provide a public library with API stability that distros can package and we can use like most of our dependencies. It still wasn't what we wanted, but now we have more experience in WebKit maintaining third party dependencies inside the source tree like ANGLE and libwebrtc, so it was no longer a blocker either. In December 2023 we made the decision of giving Skia a try internally and see if it would be worth the effort of maintaining the project as a third party module inside WebKit. In just one month we had implemented enough features to be able to run all MotionMark tests. The results in the desktop were quite impressive, getting double the score of MotionMark global result. We still had to do more tests in embedded devices which are the actual target of WPE, but it was clear that, at least in the desktop, with this very initial implementation that was not even optimized (we kept our current architecture that is optimized for CPU rendering) we got much better results. We decided that Skia was the option, so we continued working on it and doing more tests in embedded devices. In the boards that we tried we also got better results than CPU rendering, but the difference was not so big, which means that with less powerful GPUs and with our current architecture designed for CPU rendering we were not that far from CPU rendering. That's the reason why we managed to keep WPE competitive in embeeded devices, but Skia will not only bring performance improvements, it will also simplify the code and will allow us to implement new features . So, we had enough data already to make the final decision of going with Skia. In February 2024 we reached a point in which our Skia internal branch was in an "upstreamable" state, so there was no reason to continue working privately. We met with several teams from Google, Sony, Apple and Red Hat to discuss with them about our intention to switch from Cairo to Skia, upstreaming what we had as soon as possible. We got really positive feedback from all of them, so we sent an email to the WebKit developers mailing list to make it public. And again we only got positive feedback, so we started to prepare the patches to import Skia into WebKit, add the CMake integration and the initial Skia implementation for the WPE port that already landed in main. We will continue working on the Skia implementation in upstream WebKit, and we also have plans to change our architecture to better support the GPU rendering case in a more efficient way. We don't have a deadline, it will be ready when we have implemented everything currently supported by Cairo, we don't plan to switch with regressions. We are focused on the WPE port for now, but at some point we will start working on GTK too and other ports using cairo will eventually start getting Skia support as well.

- Michael Meeks: 2024-02-18 Sunday (2024/02/18 21:00)
  Up late, All Saints in the morning; nice to have Lydia playing bass; Lunch, rested, out for a walk, J. picked babes up; studied Matthew 5 in more detail. Bed.
- Federico Mena-Quintero: Rustifying libipuz: character sets (2024/02/18 04:21)
  It has been, what, like four years since librsvg got fully rustified, and now it is time to move another piece of critical infrastructure to a memory-safe language. I'm talking about libipuz, the GObject-based C library that GNOME Crosswords uses underneath. This is a library that parses the ipuz file format and is able to represent various kinds of puzzles. Libipuz is an interesting beast. The ipuz format is JSON with a lot of hair: it needs to represent the actual grid of characters and their solutions, the grid's cells' numbers, the puzzle's clues, and all the styling information that crossword puzzles can have (it's more than you think!). { "version": "http://ipuz.org/v2", "kind": ["http://ipuz.org/crossword#1", "https://libipuz.org/barred#1"], "title": "Mephisto No 3228", "styles": { "L": {"barred": "L" }, "T": {"barred": "T" }, "TL": {"barred": "TL" } }, "puzzle": [[1, 2, 0, 3, 4, {"cell": 5, "style": "L"}, 6, 0, 7, 8, 0, 9], [0, {"cell": 0, "style": "L"}, {"cell": 10, "style": "TL"}, 0, 0, 0, 0, {"cell": 0, "style": "T"}, 0, 0, {"cell": 0, "style": "T"}, 0] # the rest is omitted], "clues": { "Across": [{"number":1, "clue":"Having kittens means losing heart

for home day", "enumeration":"5", "cells":[[0,0],[1,0],[2,0],[3,0],[4,0]] }, {"number":5, "clue":"Mostly allegorical poet on writing companion poem, say", "enumeration":"7", "cells":[[5,0],[6,0],[7,0],[8,0],[9,0],[10,0],[11,0]] },] # the rest is omitted } } Libipuz uses json-glib, which works fine to ingest the JSON into memory, but then it is a complete slog to distill the JSON nodes into C data structures. You need iterate through each node in the JSON tree and try to fit its data into yours. Get me the next node. Is the node an array? Yes? How many elements? Allocate my own array. Iterate the node's array. What's in this element? Is it a number? Copy the number to my array. Or is it a string? Do I support that, or do I throw an error? Oh, don't forget the code to meticulously free the partially-constructed thing I was building. This is not pleasant code to write and test. Ipuz also has a few mini-languages within the format, which live inside string properties. Parsing these in C unpleasant at best. Differences from librsvg While librsvg has a very small GObject-based API, and a medium-sized library underneath, libipuz has a large API composed of GObjects, boxed types, and opaque and public structures. Using libipuz involves doing a lot of calls to its functions, from loading a crossword to accessing each of its properties via different functions. I want to use this rustification as an exercise in porting a moderately large C API to Rust. Fortunately, libipuz does have a good test suite that is useful from the beginning of the port. Also, I want to see what sorts of idioms appear when exposing things from Rust that are not GObjects. Mutable, opaque structs can just be passed as a pointer to a heap allocation, i.e. a Box<T>. I want to take the opportunity to make more things in libipuz immutable; currently it has a bunch of reference-counted, mutable objects, which are fine in single-threaded C, but decidedly not what Rust would prefer. For librsvg it was very beneficial to be able to notice parts of objects that remain immutable after construction, and to distinguish those parts from the mutable ones that change when the object goes through its lifetime. Let's begin! In the ipuz format, crosswords have a character set or charset: it is the set of letters that appear in the puzzle's solution. Internally, GNOME Crosswords uses the charset as a histogram of letter counts for a particular puzzle. This is useful information for crossword authors. Crosswords uses the histogram of letter counts in various important algorithms, for example, the one that builds a database of words usable in the crosswords editor. That database has a clever format which allows answering questions like the following quickly: What words in the database match ?OR?? — WORDS and CORES will match. IPuzCharset is one of the first pieces of code I worked on in Crosswords, and it later got moved to libipuz. Originally it didn't even keep a histogram of character counts; it was just an ordered set of characters that could answer the question, "what is the index of the character ch within the ordered set?". I implemented that ordered set with a GTree, a balanced binary tree. The keys in the key/value tree were the characters, and the values were just unused. Later, the ordered set was turned into an actual histogram with character counts: keys are still characters, but each value is now a count of the coresponding character. Over time, Crosswords started using IPuzCharset for different purposes. It is still used while building and accessing the database of words; but now it is also used to present statistics in the crosswords editor, and as part of the engine in an acrostics generator. In particular, the acrostics generator has been running into some performance problems with IPuzCharset. I wanted to take the port to Rust as an opportunity to change the algorithm and make it faster. Refactoring into mutable/immutable stages IPuzCharset started out with these basic operations: /* Construction; memory management */ IPuzCharset *ipuz_charset_new (void); IPuzCharset *ipuz_charset_ref (IPuzCharset *charet); void ipuz_charset_unref (IPuzCharset *charset); /* Mutation */ void ipuz_charset_add_text (IPuzCharset *charset, const char *text); gboolean ipuz_charset_remove_text (IPuzCharset *charset, const char *text); /* Querying */ gint ipuz_charset_get_char_index (const IPuzCharset *charset, gunichar c); guint ipuz_charset_get_char_count (const IPuzCharset *charset, gunichar c); gsize ipuz_charset_get_n_chars (const IPuzCharset *charset); gsize ipuz_charset_get_size (const IPuzCharset *charset); All of those are implemented in terms of the key/value binary tree that stores a character in each node's key, and a count in the node's value. I read the code in Crosswords that uses the ipuz_charset_*() functions and noticed that in every case, the code first constructs and populates the charset using ipuz_charset_add_text(), and then doesn't modify it anymore — it only does queries afterwards. The only place that uses ipuz_charset_remove_text() is the acrostics generator, but that one doesn't do any queries later: it uses the remove_text() operation as part of another algorithm, but only that. So, I thought of doing this: Split things into a mutable IPuzCharsetBuilder that has the add_text / remove_text operations, and also has a build() operation that consumes the builder and produces an immutable IPuzCharset. IPuzCharset is immutable; it can only be queried. IPuzCharsetBuilder can work with a hash table, which turns the "add a character" operation from O(log n) to O(1) amortized. build() is O(n) on the number of unique characters and is only done once per charset. Make IPuzCharset work with a different hash table that also allows for O(1) operations. Basics of IPuzCharsetBuilder IPuzCharsetBuilder is mutable, and it can live on the Rust side as a Box<T> so it can present an opaque pointer to C. #[derive(Default)] pub struct CharsetBuilder { histogram: HashMap<char, u32>, } // IPuzCharsetBuilder *ipuz_charset_builder_new (void); */ #[no_mangle] pub unsafe extern "C" fn ipuz_charset_builder_new() -> Box<CharsetBuilder> { Box::new(CharsetBuilder::default()) } For extern "C", Box<T> marshals as a pointer. It's nominally what one would get from malloc(). Then, simple functions to create the character counts: impl CharsetBuilder { /// Adds `text`'s character counts to the histogram. fn add_text(&mut self, text: &str) { for ch in text.chars() { self.add_character(ch); } } /// Adds a single character to the histogram. fn add_character(&mut self, ch: char) { self.histogram .entry(ch) .and_modify(|e| *e += 1) .or_insert(1); } } The C API wrappers: use std::ffi::CStr; // void ipuz_charset_builder_add_text (IPuzCharsetBuilder *builder, const char *text); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_builder_add_text(builder: &mut CharsetBuilder, text: *const c_char,) { let text = CStr::from_ptr(text).to_str().unwrap(); builder.add_text(text); } CStr is our old friend that takes a char * and can wrap it as a Rust &str after validating it for UTF-8 and finding its length. Here, the unwrap() will panic if the passed string is not UTF-8, but that's what we want; it's the equivalent of an assertion that what was passed in is indeed UTF-8. // void ipuz_charset_builder_add_character (IPuzCharsetBuilder *builder, gunichar ch); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_builder_add_character(builder: &mut CharsetBuilder, ch: u32) { let ch = char::from_u32(ch).unwrap(); builder.add_character(ch); } Somehow, the glib-sys crate doesn't have gunichar, which is just a guint32 for a Unicode code point. So, we take in a u32, and check that it is in the appropriate range for Unicode code points with char::from_u32(). Again, a panic in the unwrap() means that the passed number is out of range; equivalent to an assertion. Converting to an immutable IPuzCharset pub struct Charset { /// Histogram of characters and their counts plus derived values. histogram: HashMap<char, CharsetEntry>, /// All the characters in the histogram, but in order. ordered: String, /// Sum of all the counts of all the characters. sum_of_counts: usize, } /// Data about a character in a `Charset`. The "value" in a key/value pair where the "key" is a character. #[derive(PartialEq)] struct CharsetEntry { /// Index of the character within the `Charset`'s ordered version. index: u32, /// How many of this character in the histogram. count: u32, } impl CharsetBuilder { fn build(self) -> Charset { // omitted for brevity; consumes `self` and produces a `Charset` by adding // the counts for the `sum_of_counts` field, and figuring out the sort // order into the `ordered` field. } } Now, on the C side, IPuzCharset is meant to also be immutable and reference-counted. We'll use Arc<T> for such structures. One cannot return an Arc<T> to C code; it must first be converted to a pointer with Arc::into_raw(): // IPuzCharset *ipuz_charset_builder_build (IPuzCharsetBuilder *builder); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_builder_build(builder: *mut CharsetBuilder,) -> *const Charset { let builder = Box::from_raw(builder); // get back the Box from a pointer let charset = builder.build(); // consume the builder and free it Arc::into_raw(Arc::new(charset)) // Wrap the charset in Arc and get a pointer } Then, implement ref() and unref(): // IPuzCharset *ipuz_charset_ref (IPuzCharset *charet); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_ref(charset: *const Charset) -> *const Charset { Arc::increment_strong_count(charset); charset } // void ipuz_charset_unref (IPuzCharset *charset); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_unref(charset: *const Charset) { Arc::decrement_strong_count(charset); } The query functions need to take a pointer to what really is the Arc<Charset> on the Rust side. They reconstruct the Arc with Arc::from_raw() and wrap it in ManuallyDrop so that the Arc doesn't lose a reference count when the function exits: // gsize ipuz_charset_get_n_chars (const IPuzCharset *charset); #[no_mangle] pub unsafe extern "C" fn ipuz_charset_get_n_chars(charset: *const Charset) -> usize { let charset = ManuallyDrop::new(Arc::from_raw(charset)); charset.get_n_chars() } Tests The C tests remain intact; these let us test all the #[no_mangle] wrappers. The Rust tests can just be for the internals, simliar to this: #[test] fn supports_histogram() { let mut builder = CharsetBuilder::default(); let the_string = "ABBCCCDDDDEEEEEFFFFFFGGGGGGG"; builder.add_text(the_string); let charset = builder.build(); assert_eq!(charset.get_size(), the_string.len()); assert_eq!(charset.get_char_count('A').unwrap(), 1); assert_eq!(charset.get_char_count('B').unwrap(), 2); assert_eq!(charset.get_char_count('C').unwrap(), 3); assert_eq!(charset.get_char_count('D').unwrap(), 4); assert_eq!(charset.get_char_count('E').unwrap(), 5); assert_eq!(charset.get_char_count('F').unwrap(), 6); assert_eq!(charset.get_char_count('G').unwrap(), 7); assert!(charset.get_char_count('H').is_none()); } Integration with the build system Libipuz uses meson, which is not particularly fond of cargo. Still, cargo can be used from meson with a wrapper script and a bit of easy hacks. See the merge request for details. Further work I've left the original C header file ipuz-charset.h intact, but ideally I'd like to automatically generate the headers from Rust with cbindgen. Doing it that way lets me check that my assumptions of the extern "C" ABI are correct ("does foo: &mut Foo appear as Foo *foo on the C side?"), and it's one fewer C-ism to write by hand. I need to see what to do about inline documentation; gi-docgen can consume C header files just fine, but I'm not yet sure about how to make it work with generated headers from cbindgen. I still need to modify the CI's code coverage scripts to work with the mixed C/Rust codebase. Fortunately I can copy those incantations from librsvg. Is it faster? Maybe! I haven't benchmarked the acrostics generator yet. Stay tuned!

- Juan Pablo Ugarte: Cambalache Gtk4 port goes beta! (2024/02/16 14:32)
  Hi, I am happy to announce Cambalache’s Gtk4 port has a beta release! Version 0.17.2 features minors improvements and a brand new UI ported to Gtk 4! The port was easier than expected, still lots of changes as you can see here… 64 files changed, 2615 insertions(+), 2769 deletions(-) I specially like the new GtkDialog API and the removal of gtk_dialog_run() With so many changes I expect some new bugs so please if you find any file them here. Where to get it? You can get it from Flathub Beta flatpak remote-add --if-not-exists flathub-beta https://flathub.org/beta-repo/flathub-beta.flatpakrepo flatpak install flathub-beta ar.xjuan.Cambalache or checkout main branch at gitlab git clone https://gitlab.gnome.org/jpu/cambalache.git Matrix channel Have any question? come chat with us at #cambalache:gnome.org Mastodon Follow me in Mastodon @xjuan to get news related to Cambalache development. Happy coding!
- Sam Thursfield: Status update, 16/02/2024 (2024/02/16 14:16)
  Some months you work very hard and there is little to show for any of it… so far this is one of those very draining months. I’m looking forward to spring, and spending less time online and at work. Rather than ranting I want to share a couple of things from elsewhere in the software world to show what we should be aiming towards in the world of GNOME and free operating systems. Firstly, from “Is something bugging you?”: Before we even started writing the database, we first wrote a fully-deterministic event-based network simulation that our database could plug into. … if one particular simulation run found a bug in our application logic, we could run it over and over again with the same random seed, and the exact same series of events would happen in the exact same order. That meant that even for the weirdest and rarest bugs, we got infinity “tries” at figuring it out, and could add logging, or do whatever else we needed to do to track it down. The text is hyperbolic, and for some reason they

think it’s ok to work with Palantir, but its an inspiring read. Secondly, a 15 minute video which you should watch in its entirety, and then consider how the game development world got so far ahead of everyone else. Here’s a link to the video. This is the world that’s possible for operating systems, if we focus on developer experience and integration testing across the whole OS. And GNOME OS + openQA is where I see some of the most promising work happening right now. Of course we’re a looooong way from this promised land, despite the progress we’ve made in the last 10+ years. Automated testing of the OS is great, but we don’t always have logs (bug), the image randomly fails to start (bug), the image takes hours to build, we can’t test merge requests (bug), testing takes 15+ minutes to run, etc. Some of these issues seem intractable when occasional volunteer effort is all we have. Imagine a world where you can develop and live-deploy changes to your phone and your laptop OS, exhaustively test them in CI, step backwards with a debugger when problems arise – this is what we should be building in the tech industry. A few teams are chipping away at this vision – in the Linux world, GNOME Builder and the Fedora Atomic project spring to mind and I’m sure there are more . Anyway, what happened last month? Outreachy This is the final month of the Outreachy internship that I’m running around end-to-end testing of GNOME. We already had some wins, there are now 5 separate testsuites running against GNOME OS, unfortunately rather useless at present due to random startup failures. I spent a *lot* of time working with Tanju on a way to usefully test GNOME on Mobile. I haven’t been able to follow this effort closely beyond seeing a few demos and old blogs. This week was something of a crash course on what there is. Along the way I got pretty confused about scaling in GNOME Shell – it turns out there’s currently a hardcoded minimum screen size, and upstream Mutter will refuse to scale the display below a certain size. In fact upstream GNOME Shell doesn’t have any of the necessary adaptations for use in a mobile form factor. We really need a “GNOME OS Mobile” VM image – here’s an open issue – it’s unlikely to be done within the last 2 weeks of the current internship, though. The best we can do for now is test the apps on a regular desktop screen, but with the window resized to 360×720 pixels. On the positive side, hopefully this has been a useful journey for Tanju and Dorothy into the inner workings of GNOME. On a separate note, we submitted a workshop on openQA testing to GUADEC in Denver, and if all goes well with travel sponsorship and US VISA applications, we hope to actually meet in person there in July. FOSDEM I went to FOSDEM 2024 and had a great time, it was one of my favourite FOSDEM trips. I managed to avoid the ‘flu – i think wearing a mask on the plane is the secret. From Codethink we were 41 people this year – probably a new record. I went a day early to join in the end of the GTK hackfest, and did a little work on the Tiny SPARQL database, formerly known as Tracker SPARQL. Together with Carlos we fixed breakage in the CLI, improved HTTP support and prototyped a potential internship project to add a web based query editor. My main goal at FOSDEM was to make contact with other openQA users and developers, and we had some success there. Since then i’ve hashed out a wishlist for openQA for GNOME’s use cases, and we’re aiming to set up an open, monthly call where different QA teams can get together and collaborate on a realistic roadmap. I saw some great talks too, the “Outreachy 1000 Interns” talk and the Fairphone “Sustainable and Longlasting Phones” were particular highlights. I went to the Bytenight event for the first time and and found an incredible 1970’s Wurlitzer transistor organ in the “smoking area” of the HSBXL hackspace, also beating Javi, Bart and Adam at Super Tuxcart several times.

- Andy Wingo: guix on the framework 13 amd (2024/02/16 10:23)
  I got a new laptop! It’s a Framework 13 AMD: 8 cores, 2 threads per core, 64 GB RAM, 3:2 2256×1504 matte screen. It kicks my 5-year-old Dell XPS 13 in the pants, and I am so relieved to be back to a matte screen. I just got it up and running with Guix, which though easier than past installation experiences was not without some wrinkles, so here I wanted to share a recipe for what worked for me.(I swear this isn’t going to become a product review blog, but when I went to post something like this on the Framework forum I got an error saying that new users could

only post 2 links. I understand how we got here but hoo, that is a garbage experience!)The basic dealUpstream Guix works on the Framework 13 AMD, but only with software rendering and no wifi, and I wasn't able to install from upstream media. This is mainly because Guix uses a modified kernel and doesn't include necessary firmware. There is a third-party nonguix repository that defines packages for the vanilla Linux kernel and the linux-firmware collection; we have to use that repo if we want all functionality.Of course having the firmware be user-hackable would be better, and it would be better if the framework laptop used parts with free firmware. Something for a next revision, hopefully.On firmwareAs an aside, I think the official Free Software Foundation position on firmware is bad praxis. To recall, the idea is that if a device has embedded software (firmware) that can be updated, but that software is in a form that users can't modify, then the system as a whole is not free software. This is technically correct but doesn't logically imply that the right strategy for advancing free software is to forbid firmware blobs; you have a number of potential policy choices and you have to look at their expected results to evaluate which one is most in line with your goals.Bright lines are useful, of course; I just think that with respect to free software, drawing that line around firmware is not interesting. To illustrate this point, I believe the current FSF position is that if you can run e.g. a USB ethernet adapter without installing non-free firmware, then it is kosher, otherwise it is haram. However many of these devices have firmware; it's just that you aren't updating it. So for example the the USB Ethernet adapter I got with my Dell system many years ago has firmware, therefore it has bugs, but I have never updated that firmware because that's not how we roll. Or, on my old laptop, I never updated the CPU microcode, despite spectre and meltdown and all the rest."Firmware, but never updated" reminds me of the wires around some New York neighborhoods that allow orthodox people to leave the house on Sabbath; useful if you are of a given community and enjoy the feeling of belonging, but I think even the faithful would see it as a hack. It is like how Richard Stallman wouldn't use travel booking web sites because they had non-free JavaScript, but would happily call someone on the telephone to perform the booking for him, using those same sites. In that case, the net effect on the world of this particular bright line is negative: it does not advance free software in the least and only adds overhead. Privileging principle over praxis is generally a losing strategy.InstallationFirstly I had to turn off secure boot in the bios settings; it's in "security".I wasn't expecting wifi to work out of the box, but for some reason the upstream Guix install media was not able to configure the network via the Ethernet expansion card nor an external USB-C ethernet adapter that I had; stuck at the DHCP phase. So my initial installation attempt failed.Then I realized that the nonguix repository has installation media, which is the same as upstream but with the vanilla kernel and linux-firmware. So on another machine where I had Guix installed, I added the nonguix channel and built the installation media, via guix system image -t iso9660 nongnu/system/install.scm. That gave me a file that I could write to a USB stick.Using that installation media, installing was a breeze.However upon reboot, I found that I had no wifi and I was using software rendering; clearly, installation produced an OS config with the Guix kernel instead of upstream Linux. Happily, at this point the ethernet expansion card was able to work, so connect to wired ethernet, open /etc/config.scm, add the needed lines as described in the operating-system part of the nonguix README, reconfigure, and reboot. Building Linux takes a little less than an hour on this machine.Fractional scalingAt that point you have wifi and graphics drivers. I use GNOME, and things seem to work. However the screen defaults to 200% resolution, which makes everything really big. Crisp, pretty, but big. Really you would like something in between? Or that the Framework ships a higher-resolution screen so that 200% would be a good scaling factor; this was the case with my old Dell XPS 13, and it worked well. Anyway with the Framework laptop, I wanted 150% scaling, and it seems these days that the way you have to do this is to use Wayland, which Guix does not yet enable by default.So you go into config.scm again, and change where it says %desktop-services to be:(modify-services %desktop-services (gdm-service-type config => (gdm-configuration (inherit config) (wayland? #t)))) Then when you reboot you are in Wayland. Works fine, it seems. But then you have to go and enable an experimental mutter setting; install dconf-editor, run it, search for keys with “mutter” in the name, find the “experimental settings” key, tell it to not use the default setting, then click the box for “scale-monitor-framebuffer”.Then! You can go into GNOME settings and get 125%, 150%, and so on. Great.HOWEVER, and I hope this is a transient situation, there is a problem: in GNOME, applications that aren’t native Wayland apps don’t scale nicely. It’s like the app gets rendered to a texture at the original resolution, which then gets scaled up in a blurry way. There aren’t so many of these apps these days as most things have been ported to be Wayland-capable, Firefox included, but Emacs is one of them :(However however! If you install the emacs-pgtk package instead of emacs, it looks better. Not perfect, but good enough. So that’s where I am.BugsThe laptop hangs on reboot due to this bug, but that seems a minor issue at this point. There is an ongoing tracker discussion on the community forum; like other problems in that thread, I hope that this one resolves itself upstream in Linux over time.Other things?I didn’t mention the funniest thing about this laptop: it comes in pieces that you have to put together :) I am not so great with hardware, but I had no problem. The build quality seems pretty good; not a MacBook Air, but then it’s also user-repairable, which is a big strong point. It has these funny extension cards that slot into the chassis, which I have found to be quite amusing.I haven’t had the machine for long enough but it seems to work fine up to now: suspend, good battery use, not noisy (unless it’s compiling on all 16 threads), graphics, wifi, ethernet, good compilation speed. (I should give compiling LLVM a go; that’s a useful workload.) I don’t have bluetooth or the fingerprint reader working yet; I give it 25% odds that I get around to this during the lifetime of this laptop :)Until next time, happy hacking!

- Felix Häcker: #135 Experimental Maps (2024/02/16 00:00)
  Update on what happened across the GNOME project in the week from February 09 to February 16. Sovereign Tech Fund Tobias Bernard announces As part of the GNOME STF (Sovereign Tech Fund) project, a number of community members are working on infrastructure related projects. Accessibility Joanie worked on De-PyAtSpi-ing Orca AtspiDocument: done Create AT-SPI2 based utilities for the document interface.: https://gitlab.gnome.org/GNOME/orca/-/commit/f36310b6e Use AXDocument in chromium, gecko, and thunderbird scripts + bookmarks: https://gitlab.gnome.org/GNOME/orca/-/commit/ebe677317 Use AXDocument in the web script: https://gitlab.gnome.org/GNOME/orca/-/commit/7a3dc4871 AtspiComponent: done Add an AT-SPI2 grab_focus function to AXObject and use it instead of pyatspi: https://gitlab.gnome.org/GNOME/orca/-/commit/a43dfffa3 Create AT-SPI2 component interface utilities: https://gitlab.gnome.org/GNOME/orca/-/commit/47c76e406 Use AXComponent in the event synthesizer: https://gitlab.gnome.org/GNOME/orca/-/commit/1e004fe98 Use AXComponent in more places: https://gitlab.gnome.org/GNOME/orca/-/commit/a0602b147 Begin using AXComponent in scripts and Where Am I Presenter: https://gitlab.gnome.org/GNOME/orca/-/commit/2e595ce6a Use AXComponent in script utilities and flat review: https://gitlab.gnome.org/GNOME/orca/-/commit/b6a87ff05 Use AXComponent in a couple of places missed before: https://gitlab.gnome.org/GNOME/orca/-/commit/c1761de2a Remove code obsoleted by AXComponent: https://gitlab.gnome.org/GNOME/orca/-/commit/1573c7d37 AtspiText: WIP Last (known) use of pyatspi in Orca Orca should be pyatspi-free when Orca v46 is released ▯ Joanie removed some hacks from Orca Remove broken _adjustPositionForObj hack: https://gitlab.gnome.org/GNOME/orca/-/commit/1654dfa69 Remove the isShowingAndVisible hack: https://gitlab.gnome.org/GNOME/orca/-/commit/e857c5361 Remove isZombie; use AXObject.is_valid instead: https://gitlab.gnome.org/GNOME/orca/-/commit/5c3ef549f Web: Eliminate (nearly all of) our text sanity-checks and hacks: https://gitlab.gnome.org/GNOME/orca/-/commit/0aab0e5da Sam fixed some minor things in GNOME Shell Style papercut fixes before freeze:

https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/3165 Fixed the Screen Reader regression in Shell’s Alt-Tab interface: https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/3127 Andy continued work in integration Spiel with Orca https://github.com/eeejay/spiel-demos/pull/1 https://gitlab.gnome.org/GNOME/orca/-/merge_requests/182 New Accessibility Stack (Newton) Matt is working on a client library for Orca and other ATs Platform Andy fixed some papercuts withthe gnome-online-accounts settings panel UI: https://gitlab.gnome.org/GNOME/gnome-control-center/-/issues/2891 Julian worked on making notifications in the calendar popover expandable and worked with Sam on refreshing the style https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/3173 Andy, Tobias, and Sam worked on fixing papercuts with the gnome-online-accounts GTK4 port https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/162 https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/164 https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/175 We are investigating a solution for a UX regression AdwDialog https://gitlab.gnome.org/GNOME/libadwaita/-/issues/797 Evan landed async support on gobject introspection https://gitlab.gnome.org/GNOME/gobject-introspection/-/merge_requests/404 Async API support in GLib is deferred to next release https://gitlab.gnome.org/GNOME/glib/-/merge_requests/3746 Looking at how to update meson in other projects to use the new GLib GI compiler instead of the old one without async support. Evan is working on finishing merging the gi.ts and ts-for-gir codebases for GI-driven TypeScript bindings. MR here: https://github.com/gjsify/ts-for-gir/pull/144 Sonny landed AdwDialog support in Blueprint https://gitlab.gnome.org/jwestman/blueprint-compiler/-/merge_requests/177 Andy landed GTK4 port of GNOME Online Account https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/142 https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2039 Andy landed migratation of existing WebDAV accounts in GOA https://gitlab.gnome.org/GNOME/gnome-online-accounts/-/merge_requests/146 Hardware Support Jonas worked on fractional scaling on XWayland https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3567 Jonas opened a mutter MR for fractional scaling on XWayland https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3567 Jonas landed screencast pipeline blocklisting (https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/2976) we are very close to getting the HW encoder support merged https://gitlab.gnome.org/GNOME/gnome-shell/-/merge_requests/2080 Dor worked on variable refresh rate support, which is still under review https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/1154 Dor implemented small refinements to the Settings UI for VRR https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/734 Dor added support for showing the VRR range of monitors in the Settings UI when possible: Mutter: https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/3576 Settings: https://gitlab.gnome.org/GNOME/gnome-control-center/-/merge_requests/2260 Sam and Dor continued to iterate on the design for VRR in display settings adjusted the design for the proposed list rows: https://gitlab.gnome.org/GNOME/gnome-control-center/-/issues/2523#note_2005469 Security Dhanuka continued work on implementing the oo7-daemon: https://github.com/bilelmoussaoui/oo7/pull/56 Fixed most of the issues (a few remains) flagged by Bilal’s review. Fixed some clippy warnings found in oo7 client when the unstable feature is on: https://github.com/bilelmoussaoui/oo7/pull/67 Updated oo7::portal::Keyring::create_item() to return portal::Item: https://github.com/bilelmoussaoui/oo7/pull/68 Updated oo7::portal::Keyring API’s “attributes” parameter: https://github.com/bilelmoussaoui/oo7/pull/69 Wayland & Portal APIs Hubert worked on a number of flatpak/portal related things: libportal settings support Fixed test so the app fits on smaller screens: https://github.com/flatpak/libportal/pull/142 Settings API (in review) https://github.com/flatpak/libportal/pull/143 flatpak device support: Proposed and started implementing a compatibility mechanism: https://github.com/flatpak/flatpak/issues/5681 Running the test with ASAN leak all over the place: PR https://github.com/flatpak/flatpak/pull/5683 GNOME Core Apps and Libraries Maps Maps gives you quick access to maps all across the world. James Westman reports Maps' “Experimental Map” mode has a new look for GNOME 46! Features include: Dark mode Translated labels, where available Large Text accessibility setting Symbols for highway routes Adwaita icons You can read more about the new style in my blog post. Also in experimental mode, you can now click a label and the info bubble appears immediately, no need to right click and choose “What’s Here?”. GNOME Circle Apps and Libraries zeenix says zbus 4.0 released. zbus is a pure Rust D-Bus crate that many GNOME apps and components rely on. The new version brings a more ergonomic and safer API. NewsFlash feed reader Follow your favorite blogs & news sites. Jan Lukas says Newsflash 3.1 was released. It brings quality of life improvements and bugfixes. https://blogs.gnome.org/jangernert/2024/02/12/newsflash-3-1/ Apostrophe A distraction free Markdown editor. Manu says Apostrophe’s bottombar has seen some improvements. Now the statistics button adapts to the available space, and a lot of under-the-hood changes have been made to the whole bottombar widget Your browser doesn't support embedded videos, but don't worry, you can download it and watch it with your favorite video player! Third Party Projects ghani1990 reports This week, Flashcards app has introduced some exciting changes that enhance user experience and organization: Tags: Now, users can group flashcards within a set using tags. This feature streamlines organization and makes it easier to manage related cards. Keywords: The app also introduces keywords, allowing users to group entire sets. These keywords serve as filters in the sidebar, making it effortless to locate specific sets. دانیال بهزادی says Love is in the air for privacy enthusiasts and individuals residing in tyrannical states. As a Valentine’s Day gift version 4.5.0 of Carburetor released with support for Snowflake and WebTunnel bridges, providing you with even more opportunities to seamlessly connect to the TOR network. Built upon Libadwaita, Carburetor allows you to easily set up a TOR proxy on your session without getting your hands dirty with system configs. Initially designed to simplify the lives of GNOME enthusiasts on their mobile phones, it’s now fully usable with mouse and/or keyboard too. ghani1990 says A new update for Done,The ultimate task management solution for seamless organization and efficiency, is now available! This update includes a new design, a moved services bar, and the ability to expand task details from within each task. Dexter Reed announces Chats (aka Chatty) has been ported to use the modern libadwaita 1.4 widgets and sidebar by Chris Talbot, with some help from me (sungsphinx) and Bart Gravendeel (aka Monster). You can view the pull request here: https://gitlab.gnome.org/World/Chatty/-/merge_requests/1317 xjuan says I am happy to announce Cambalache’s Gtk4 port has a beta release! Version 0.17.2 features minors improvements and a brand new UI ported to Gtk 4! Available in flathub beta! More information at https://blogs.gnome.org/xjuan/2024/02/16/cambalache-gtk4-port-goes-beta/ Blueprint A markup language for app developers to create GTK user interfaces. Sonny announces AdwAlertDialog support was added to Blueprint. Blueprint is able to support new widgets automatically thanks to GObject Introspection but some widgets have custom builder syntax that needs to be added to Blueprint. Here is the merge request if you’d like to see how custom builder syntax works in Blueprint https://gitlab.gnome.org/jwestman/blueprint-compiler/-/merge_requests/177 That’s all for this week! See you next week, and be sure to stop by #thisweek:gnome.org with updates on your own projects!

- Jiri Eischmann: Bisecting Regressions in Fedora Silverblue (2024/02/15 10:00)
  I know that some blog posts on this topic have already been published, but nevertheless I have decided to share my real-life experience of bisecting regressions in Fedora Silverblue. My work laptop, Dell XPS 13 Plus, has an Intel IPU6 webcam. It still lacks upstream drivers, so I have to use RPMFusion packages containing Intel’s software stack to ensure the webcam’s functionality. However, on Monday, I discovered that the

webcam was not working. Just last week, it was functioning, and I had no idea what could have broken it. I don't pay much attention to updates; they're staged automatically and applied with the next boot. Fortunately, I use Fedora Silverblue, where problems can be identified using the bisection method. Silverblue utilizes OSTree, essentially described as Git for the operating system. Each Fedora version is a branch, and individual snapshots are commits. You update the system by moving to newer commits and upgrade by switching to the branch with the new version. Silverblue creates daily snapshots, and since OSTree allows you to revert to any commit in the repository, you can roll back the system to any previous date. I decided to revert to previous states and determine when the regression occurred. I downloaded commit metadata for the last 7 days: sudo ostree pull --commit-metadata-only --depth 7 fedora fedora/39/x86_64/silverblue Then, I listed the commits to get their labels and hashes: sudo ostree log fedora:fedora/39/x86_64/silverblue Subsequently, I returned the system to the beginning of the previous week: rpm-ostree deploy 39.20240205.0 After rebooting, I found that the webcam was working, indicating that something had broken it last week. I decided to halve the interval and return to Wednesday: rpm-ostree deploy 39.20240207.0 In the Wednesday snapshot, the webcam was no longer functioning. Now, I only needed to determine whether it was broken by Tuesday's or Wednesday's update. I deployed the Tuesday snapshot: rpm-ostree deploy 39.20240206.0 I found out that the webcam was still working in this snapshot. It was broken by the Wednesday's update, so I needed to identify which packages had changed. I used the hashes from one of the outputs above: rpm-ostree db diff ec2ea04c87913e2a69e71afbbf091ca774bd085530bd4103296e4621a98fc835 fc6cf46319451122df856b59cab82ea4650e9d32ea4bd2fc5d1028107c7ab912 ostree diff commit from: ec2ea04c87913e2a69e71afbbf091ca774bd085530bd4103296e4621a98fc835 ostree diff commit to: fc6cf46319451122df856b59cab82ea4650e9d32ea4bd2fc5d1028107c7ab912 Upgraded: kernel 6.6.14-200.fc39 -> 6.7.3-200.fc39 kernel-core 6.6.14-200.fc39 -> 6.7.3-200.fc39 kernel-modules 6.6.14-200.fc39 -> 6.7.3-200.fc39 kernel-modules-core 6.6.14-200.fc39 -> 6.7.3-200.fc39 kernel-modules-extra 6.6.14-200.fc39 -> 6.7.3-200.fc39 llvm-libs 17.0.6-2.fc39 -> 17.0.6-3.fc39 mesa-dri-drivers 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-filesystem 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-libEGL 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-libGL 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-libgbm 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-libglapi 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-libxatracker 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-va-drivers 23.3.4-1.fc39 -> 23.3.5-1.fc39 mesa-vulkan-drivers 23.3.4-1.fc39 -> 23.3.5-1.fc39 qadwaitadecorations-qt5 0.1.3-5.fc39 -> 0.1.4-1.fc39 uresourced 0.5.3-5.fc39 -> 0.5.4-1.fc39 The kernel upgrade from 6.6 to 6.7 was the logical suspect. I informed a colleague, who maintains the RPMFusion packages with the IPU6 webcams support, that the 6.7 kernel broke the webcam support for me. He asked for the model and informed me that it had an Intel VSC chip, which has a newly added driver in the 6.7 kernel. However, it conflicts with Intel's software stack in RPMFusion packages. So he asked the Fedora kernel maintainer to temporarily disable the Intel VSC driver. This change will come with the next kernel update. Until the update arrives, I decided to stay on the last functional snapshot from the previous Tuesday. To achieve this, I pinned it in OSTree. Because the system was currently running on that snapshot, all I had to do was: sudo ostree admin pin 0 And that's it. Just a small real-life example of how to identify when and what caused a regression in Silverblue and how to easily revert to a version before the regression and stay on it until a fix arrives.

- Luis Villa: Boox Page: A Review (2024/02/15 02:26)
  I love words, and have as far back as I can remember. Mom says I started reading the newspaper religiously in kindergarten, I proudly show off a small percentage of my books in my videoconferencing background, and of course the web is (still) junk straight into my veins. So I've long wanted an e-ink device that could do multi-vendor ebooks, pdfs, and a feed reader. (I still consume a lot of RSS, mostly via the excellent feedbin.)

But my 8yo Kindle kept not dying, and my last shot at this (a Sony) was expensive and frustrating. Printery, by Dennis Jarvis; used under CC BY-2.0. Late last year, though, I found out about the Boox Page. It is… pretty great? Some notes and tips: My use case: I don’t write or take notes on this, so I can’t advise on whether it is a good Remarkable (or similar) competitor. Also don’t use the music player or audiobooks. Price: Pretty reasonable for something I use this much. Open question how long they do Android support, unfortunately—Reddit users report five years for older models, which isn’t great but (unlike a phone) not as hugely active a security risk. Screen and body: screen, lighting, size, and weight are easily on par with my old Kindle, which is to say “comfortable on eyes and hands for a lot of long-term reading”. Battery life: This is not an OG Kindle, but I have gone multiple plane flights without recharging, and it is USB-C, so (unlike my micro-USB Kindle) easy to charge while traveling. Other hardware: Meh. The hardware volume buttons are… fine? But small. I would not miss them if I had to rely on the screen. The cover is a fine cover, but it is not an effective stand – something I miss from my Kindle. I suspect repairability is terrible. Sorry, iFixit :(Core operating system UX: Android modded to use e-ink is… surprisingly fine? I use the Android layer essentially to (1) install apps and (2) launch them. And those work well, though the default launcher screen has some clutter that can’t be removed. Some of the alternate Android launchers apparently work, but I haven’t tried any of them yet. performance: Boot time is very long, and for unknown reasons it seems to default to a full shutdown when you close the cover, which really destroys the “open up the book and immediately start reading” experience. Strongly recommend Power → Power-off Timeout → “never” to avoid this problem. Security: There is no thumbprint or face scan here, and the e-ink keyboard is as good as possible but still slow to enter a high-strength password. So I’ve chosen to put as little private information on this as possible. Has motivated me to update some passwords to “xkcd-style” four-word, lower-case, strings in order to make them easier to read from my phone and type in on the Boox. Openness: it looks like Onyx violates the GPL. Folks have successfully rooted older Boox devices, but I have not seen any evidence of anyone doing that with the Page yet. UX “improvements”: There are so many; most aren’t great but the worst ones can all be turned off. “Naviball”: a weird dot-as-UI thing that I turned off nearly immediately. Gestures: To save screen real-estate, standard Android back and home buttons are by default replaced with gestures (documentation). I found the gestures a bit unreliable and so turned the bottom toolbar back on. Not great but fine. E-ink settings: There are a lot of different settings for e-ink refresh strategies (eg do you want it faster but lossy, slower but clearer, etc.), which are adjustable both overall and on a per-app basis. The default settings seem to work pretty well for me, but I suspect at least some apps would benefit from tweaking. Docs here for more details. App stores: Google Play: pre-installed, and can be used to install Kindle, Libby, and Hoopla. It works fine, with two caveats: I don’t love giving this device my Google password; and it is clear that the best iOS apps have noticeably better UX than the best Android apps. F-Droid: Works great! Used it to install the Wikipedia app, KOReader, and Tusky, among others. Getting books: Kindle app works great. You will not lose access to your existing e-book library by moving away from a Kindle. Libby and Hoopla: Both of the apps used by my library (San Francisco Public) to make e-books available install fine from the Google Play App Store and work great. Have happily read long books in both of them. Standard Ebooks: Their website and “advanced epub” work great on the Boox, so I’ve been plowing through a bunch of old public domain stuff. This has been maybe the single most satisfying part of the device, to be honest—it’s been great to refocus my reading that way instead of being guided by Amazon’s recommendations. Onyx: the folks behind Boox have a book store, which can be turned off but not removed from the home-screen UX. I have not used it. Reading books: All of the apps above have worked fine, with the minor detail that in some of them (notably Kindle) you have to find a setting to enable “use volume control as page up/down” in order to use the page turning buttons. KOReader, from F-Droid, was recommended to me. Its default text presentations are, IMHO, not ideal. But once I spent some time experimenting to get proper margins and line spacing, it became very pleasant. I use it to read the “advanced” formatted epubs from

Standard Ebooks, and it handles those very well. Managing books: KO Reader and the built-in book reader mostly default to “just expose the file system” as a book management view, which is… not great. KO Reader does not sync reading positions, so if you’re used to going back and forth between your e-ink device and your phone, you may need to do more work, either to just remember your positions, or to set up something like calibre-web. I’m using Story Graph as my main “what did I read lately” repository and it works great as a progressive web app. The web The basics: the Page has a reasonably modern Chrome (called “Neobrowser”). Mostly just works, none of the “experimental” web browser stuff from Kindle. Scrolling: can be painful; e-ink and scrolling don’t mix too well. Much depends on the nature of the web page (fine) or web app (grimace) you’re using – some are fine, others are not so much. Progressive Web Apps: The browser often prompts you to install web sites as app icons on the main launcher page and they generally work great. Really nice alternative to app stores, just like we always hoped they’d be. Maybe just a few years too late :(Other sources of text Feeds: Feedbin.com in the browser is… fine. Refresh is sort of wonky, though. I’ve tried both Android apps listed on the feedbin site, but don’t love either. (I like the experience enough that I have resurrected my dream of a more newspaper-like app to read on this in the morning.) Fediverse: Tusky is the most-recommended open Android Mastodon app, and it is fine, but I have found that I prefer Phanpy in-browser (despite some performance challenges) to the better performance but rough UX edges of Tusky. There are quite a few other options; I have not explored them deeply. PDFs. The couple of PDFs I’ve tried to read have been fine, but I have not pushed it and I suspect they won’t be great given the smaller-screen; either a larger Boox or a Remarkable seem more the right thing if that is your primary use case. I really need to set up some sort of syncing and prioritizing system; many of them are reputed to work (again, basically stock-ish Android). Wikipedia: App works great. Haven’t tried editing yet though ;) Things I’d like to explore more: Security: can I move entirely to F-Droid-sourced apps so I don’t need this to have my Google password? Can I replace most Onyx-provided apps next? “Find my”: Is there a “find this device” (and/or “remote wipe this”) service that would help me track it down in case of loss, but that doesn’t require trusting the device very much (see Google password above)? No idea but given that I’ve already nearly lost it once, I need to look into it. Sharing out: I like sharing out links, quotes, snippets, etc. but typing on this is a bit of a mess. I’d like to figure out some sort of central output flow, where I can use the Android share feature to dump into a stream that I can then surf and resurface from other devices. Not sure what the right service is to host that, though. Bottom line: I am really enjoying this, and it is enough under my control that I suspect I can make it even better. It certainly isn’t a dream device from a freedom and privacy perspective, but for my limited use case that’s fine. Added, Feb. 15: An old friend reading this said it surprised him, because my software aesthetics at this point are very much “it had better work out of the box, I have no time or patience for DIY”, and this sounded… pretty DIY. It’s a great observation. For lots of other devices, I have no patience for something that doesn’t Just Work. But apparently (surprising me as much as anyone!) I have a lot of patience for getting my e-reading experience just so. So this did (still does) take patience, but also so far has rewarded that patience. We’ll see if it sticks.

- Marcus Lundblad: The FOSDEM trip 2024 (2024/02/14 22:25)
  As there was some interest and questions about my trip to Brussels and FOSDEM on Mastodon, I thought I should write down some notes and observations from the trip. This will not really be about FOSDEM itself as there's numerous other reports from the conference itself.TicketsI had some questions about how and where to buy tickets for a train journey like this.For the first connection with the commuter train to Stockholm C, I just used my regular 30 day pass for the Stockholm and Uppsala regional local traffic as it already had validity covering the entire trip.For the rest of the trip I made two separate reservations. One for a round-trip journey from Stockholm C to Hamburg Hbf in a shared couchette (six bed compartment) on the EuroNight service booked via sj.se. Leaving in the evening Feb 1 (Thursday) at 17:34, return trip from Hamburg on Feb 5

(Monday) at 22:03. I then separetly booked tickets on ICE from Hamburg Hbf to Brussel Nord via bahn.de.For the second part I choose the option with seat reservation, but bound to specific trains. Specifically departing from Hamburg at 10:45 on Feb 2. This gave me almost 5 hours margin, which is perhaps a bit on the safing side, and adds to the total journey time. On the other hand it gives some extra time to have some breakfast and walk around a bit (though it's a bit early in the morning).Return trip from Brussels to Hamburg was scheduled to arrive at 17:18, giving plenty of time (almost five hours) to get some dinner and visit some sights. Here it's more crucial to set aside time for any hickups, as missing the night train service would be pretty awkward…Another option would have been to get the InterRail pass. But bare in mind that night trains still require buying reservation. And reseving seats on German ICE might still be a good idea to ensure getting a fixed seat (would be especially beneficial if you intend to work on the train).The journeyA side note: As my camera has started acting up, taking several attempts to start up after being off for a while (probably the lens mechanism getting worn after taking around 21000 pictures), I haven't taken as much pictures as usual…I left home after lunch on the Thursday, first going for the commuter train.Trains at Uppsala CArriving at Stockholm C (techinically Stockholm City, the underground station connected to the metro system) I left suitcase in a baggage locker at the station. Now the plan was to take the tram out to Djurgården (this is also where the Vasa and ABBA musuems in Stockholm are located) to have some fika and enjoy the weather as it was one of those sunny days. But since the next tram then was a bus replacement, I decided to instead take a walk.View over Sergels torg in StockholmFinally some fika at the cafe Lilla HasselbackenThe headed back towards the central station, picket up my suitcase and went to the restaurant Belgobaren to have dinner and a couple of beers. Also a good way to warm up for a bit of Belgian spirit The nice bar at Belgobaren. This place is also the hotel restaurant for Freys HotelHalf an hour or so before the night train was about to leave I headed back to the station.The departure boardAt track 10, next train is ours…These are old German sleeper carsArriving in Hamburg the morning after around halv an hour or so behind scheduleHad a bit of breakfast at one of the cafe places in the station.Coffee and a sandwichTaking a morning walk and visiting the exhibition at the city hall (which was thankfully opening at this early time of day).Hamburh Rathaus (city hall)Arriving back at the station awaiting the departure of the next train towards Köln (Cologne).Shortly before the train was supposed to depart it was announced as being cancelled…Asked some staff from Deutche Bahn at the platform and got a suggestion to instead take a train to Düsseldorf and then on to Liège, and from there to Brussel Nord.At Düsseldorf HbfArriving in Düsseldorf and boarding the train towards Liège (destination Paris Nord), it turns out this was a Eurostar. And they did not accept my ticket even though I think that's what the staff had told me (unless I misunderstood the German). So I had to pay for a new ticket onboard.Later filed a claim for a refund for this. And as I had not registered an account beforehand at bahn.de with my e-mail address, this had to be done via a printed form and old-school mail… So this is something to keep in mind, registering your booking beforehand could be a good idea.Eventually arrived in Brussels an hour and some delayed.The actual FOSDEMNot so much from the acual FOSDEM in this post, but OK a few pictures from ULB…The journey back homeStarted off the morning on Monday Feb 5 by walking around a bit in central Brussels, saying hi to Maneken Pis, and buying some beer and chocolate.The next thing that happend was a bit eventful though…Headed back towards the northern station (Bruxelles Nord), as this was what I had booked seat for (probably wouldn't have been a problem getting on the train at the central station, but). Taking the tram from De Brouckère towards the northern station. I had about an hour to spare here. These lines (3 and 4) runs in a tunnel. This is what could be called a pre-metro). After the stop at Rogier (one stop from where I was about to get off) there was a sudden stop, and a power-outage! (ouch!).After maybe half an hour we had to evacuate walking through the tunnel back to Rogier. Then walked to station. Still in quite good time for the train (but I was getting quite nervous for a while).Later on the ICE towards Köln was some 20 min delayed. This train was supposed to continue on to Franfurt am Main, but turned back to Brussels in Köln. Fortunatly (for me) this didn't affect me, as I was getting off anyway.At Köln Hbf Got a little less time than planned to get some snack at Köln Hbf. Arriving in Hamburg, after leaving the suitcase in a baggage locker, I took the S-Bahn to Landungsbrücke to visit the submarine museum.The old Soviet submarine. Now as the U-434 submarine museum.And after that a nice dinner at Blockbräu.Blockbräu restaurantAnd then a qick a look at the old tunnel under the river Elbe, where you take elevators down. There is still construction going on and one of the tunnel lines is closed. Seems to only be opened for pedestrian and bike traffic. This was also the case when I visited last year.The old tunnelElevators taking cars. There's also smaller „regular" elevators. But I guess those are much newer…Back at the station awaiting the train.The night train to StockholmAnd the morning after looking out from the last car of the train on the long straight around MjölbyAnd then a bit later, after lunch arriving home after a nice weekend in Brussels attending FOSDEM, listening to interesting FOSS talks and hanging a bit with GNOME folks.

- Jan Lukas Gernert: Newsflash 3.1 (2024/02/12 08:55)
  (yes, you read that right: the 'f' is not capitalized anymore) This release doesn't introduce groundbreaking new functionality. But there are quite a few quality of life improvements worth checking out. Marking all articles as read is now more predictable and can be undone. It takes into account if the article list is filtered by a search term or viewing only starred articles. Only those will get marked as read. If you still accidentally mark more articles as read than intended there is a grace period that allows you to undo your action. Subscribing to a new feed used to add it to the sidebar. But no articles were fetched until the next sync. No longer: Newsflash will now fetch articles of that particular feed right after it was added. Speaking of fetching articles for feeds: if a feed fails to download or parse for some reason it will now display an icon in the sidebar. This works for local RSS and all services that propagate their errors via their API. My personal highlight in this release is the miniflux implementation supporting enclosures. This means more thumbnails and attachments for users if they run a recent version of miniflux (>= 2.0.49). Other than that a few more smaller changes and bug fixes can be found in Newsflash 3.1 Get it on flathub
- Bastien Nocera: New and old apps on Flathub (2024/02/09 14:42)
  3D Printing Slicers I recently replaced my Flashforge Adventurer 3 printer that I had been using for a few years as my first printer with a BambuLab X1 Carbon, wanting a printer that was not a "project" so I could focus on modelling and printing. It's an investment, but my partner convinced me that I was using the printer often enough to warrant it, and told me to look out for Black Friday sales, which I did. The hardware-specific slicer, Bambu Studio, was available for Linux, but only as an AppImage, with many people reporting crashes on startup, non-working video live view, and other problems that the hardware maker tried to work-around by shipping separate AppImage variants for Ubuntu and Fedora.After close to 150 patches to the upstream software (which, in hindsight, I could probably have avoided by compiling the C++ code with LLVM), I manage to "flatpak" the application and make it available on Flathub. It's reached 3k installs in about a month, which is quite a bit for a niche piece of software.Note that if you click the "Donate" button on the Flathub page, it will take you a page where you can feed my transformed fossil fuel addiction buy filament for repairs and printing perfectly fitting everyday items, rather than bulk importing them from the other side of the planet. Preparing a Game Gear consoliser shellI will continue to maintain the FlashPrint slicer for FlashForge printers, installed by nearly 15k users, although I enabled automated updates now, and will not be updating the release notes, which required manual intervention.FlashForge have unfortunately never answered my queries about making this distribution of their software official (and fixing the crash when using a VPN...). Rhythmbox As I was updating the Rhythmbox Flatpak on Flathub, I realised that it just reached 250k installs, which puts the number of installations of those 3D printing slicers above into perspective. The updated screenshot used on FlathubCongratulations, and many thanks, to all the developers that keep on contributing to this very mature project, especially Jonathan Matthew who's been maintaining

the app since 2008.

- Martín Abente Lahaye: Gameeky released (2024/02/08 13:33)
  After three months of development, Gameeky reaches its first public release. This project is the result of nearly fifteen years of experience contributing to education projects and mentoring young learners. I am happy to share it everyone! Although this project can still be considered in early stages of development, it's not far from being feature complete. This first release comes the following: A game launcher to manage projects more easily. A game client to play games cooperatively. A scene editor to create and modify game worlds. An entity editor to create and modify game objects. A Python library for a LOGO-like experience. Plugins support to extend the games logic and entities behavior. An offline beginner's guide. The first thematic pack to create role-playing games in a medieval fantasy setting. And more… Gameeky is available in English and Spanish, including the beginner's guide. The recommended installation method is through GNOME Software from Flathub. Alternatively, it can also be installed from the terminal: flatpak --user install flathub dev.tchx84.Gameeky The first thematic pack can also be installed from the terminal: flatpak --user install flathub dev.tchx84.Gameeky.ThematicPack.FreedomValley Moving forward, these are the next steps for Gameeky: Improve the user experience and accessibility, e.g., supporting undo operations on the editors and allowing these editors to be fully usable with the keyboard. Stress test the entities system, e.g., figuring out what else can't be modeled with it and improve it without making it unnecessarily complex. Address performance issues, e.g., optimizing the rendering pipeline and the communication protocol. Create new thematic packs for different interests, e.g., robotics, sci-fi, urban cities, etc. Produce more in-depth tutorials, e.g., videos to show how to create new thematic packs from scratch, an introductory course to Python built on Gameeky, etc. Support more languages, e.g., providing translations of its user interface and beginner's guide. Other important miscellaneous tasks, e.g., designing a proper icon for the desktop. To achieve some of these goals I plan to take Gameeky to its users, by organizing workshops and hackathons with students and teachers here in Paraguay. If you know any organization or individual that might be interested in supporting these workshops financially or otherwise, please feel free to contact me.
- Robert McQueen: Flathub: Pros and Cons of Direct Uploads (2024/02/06 10:57)
  I attended FOSDEM last weekend and had the pleasure to participate in the Flathub / Flatpak BOF on Saturday. A lot of the session was used up by an extensive discussion about the merits (or not) of allowing direct uploads versus building everything centrally on Flathub's infrastructure, and related concerns such as automated security/dependency scanning. My original motivation behind the idea was essentially two things. The first was to offer a simpler way forward for applications that use language-specific build tools that resolve and retrieve their own dependencies from the internet. Flathub doesn't allow network access during builds, and so a lot of manual work and additional tooling is currently needed (see Python and Electron Flatpak guides). And the second was to offer a maybe more familiar flow to developers from other platforms who would just build something and then run another command to upload it to the store, without having to learn the syntax of a new build tool. There were many valid concerns raised in the room, and I think on reflection that this is still worth doing, but might not be as valuable a way forward for Flathub as I had initially hoped. Of course, for a proprietary application where Flathub never sees the source or where it's built, whether that binary is uploaded to us or downloaded by us doesn't change much. But for an FLOSS application, a direct upload driven by the developer causes a regression on a number of fronts. We're not getting too hung up on the "malicious developer inserts evil code in the binary" case because Flathub already works on the model of verifying the developer and the user makes a decision to trust that app – we don't review the source after all. But we do lose other things such as our infrastructure building on multiple architectures, and visibility on whether the build environment or upload credentials have been compromised unbeknownst to the developer. There is now a manual review process for when apps change their

metadata such as name, icon, license and permissions – which would apply to any direct uploads as well. It was suggested that if only heavily sandboxed apps (eg no direct filesystem access without proper use of portals) were permitted to make direct uploads, the impact of such concerns might be somewhat mitigated by the sandboxing. However, it was also pointed out that my go-to example of “Electron app developers can upload to Flathub with one command” was also a bit of a fiction. At present, none of them would pass that stricter sandboxing requirement. Almost all Electron apps run old versions of Chromium with less complete portal support, needing sandbox escapes to function correctly, and Electron (and Chromium’s) sandboxing still needs additional tooling/downstream patching to run inside a Flatpak. Buh-boh. I think for established projects who already ship their own binaries from their own centralised/trusted infrastructure, and for developers who have understandable sensitivities about binary integrity such such as encryption, password or financial tools, it’s a definite improvement that we’re able to set up direct uploads with such projects with less manual work. There are already quite a few applications – including verified ones – where the build recipe simply fetches a binary built elsewhere and unpacks it, and if this already done centrally by the developer, repeating the exercise on Flathub’s server adds little value. However for the individual developer experience, I think we need to zoom out a bit and think about how to improve this from a tools and infrastructure perspective as we grow Flathub, and as we seek to raise funds for different sources for these improvements. I took notes for everything that was mentioned as a tooling limitation during the BOF, along with a few ideas about how we could improve things, and hope to share these soon as part of an RFP/RFI (Request For Proposals/Request for Information) process. We don’t have funding yet but if we have some prospective collaborators to help refine the scope and estimate the cost/effort, we can use this to go and pursue funding opportunities.

- Richard Hughes: fwupd: Auto-Quitting On Idle, Harder (2024/02/05 13:25)
  In fwupd 1.9.12 and earlier we had the following auto-quit behavior: Auto-quit on idle after 2 hours, unless: Any thunderbolt controller, thunderbolt retimer or synaptics-mst devices exist. These devices are both super slow to query and also use battery power to query as you have to power on various hungry things and then power them down to query for the current firmware version. In 19.13, due to be released in a few days time, we now: Auto-quit after 5 minutes, unless: Any thunderbolt controller, thunderbolt retimer or synaptics-mst devices exist. Any D-Bus client (that used or is using fwupd) is still alive, which includes gnome-software if it’s running in the background of the GNOME session The daemon took more than 500ms to start – on the logic it’s okay to wait 0.5 seconds on the CLI to get results to a query, but we don’t want to be waiting tens of seconds to check for updates on a deeply nested USB hub devices. The tl;dr: is that most laptop and desktop machines have Thunderbolt or MST devices, and so they already had fwupd running all the time before, and continue to have it running all the time now. Trading 3.3MB of memory and an extra process for instant queries on a machine with GBs of memory is probably worthwhile. For embedded machines like IoT devices, and for containers (that are using fwupd to update things like the dbx) fwupd was probably starting and then quitting after 2h before, and now fwupd is only going to be alive for 5 minutes before quitting. If any of the thresholds (500 ms) or timeouts (5 mins) are offensive to you then it’s all configurable, see man fwupd.conf for details. Comments welcome.
- Ismael Olea: Last activity in Wikimedia (2024/02/04 23:00)
  A fast update of thing happening to me in the Wikimedia Movement in the last weeks: We asked for a WM Rapid Grant for a new project we have conceptualized at LaOficina: SMALL GLAM SLAM Pilot 1, to low entry barriers of software and practices adoption for very small GLAM organizations. Also, I have been granted an scholarship to attend the Wikimedia Hackathon 2024 event next May in Tallinn. And, as part of my travel connection, I plan to attend the more informal GLAM + Commons + AI sauna in Helsinki. So, another Wikimedia GLAM overdose after the

excellent Montevideo’s GLAM Wiki 2023 meeting. I’ve draft the main tasks for the hackathon. They are very big in scope but they would be adjousted with the results of the Pilot 1. Here they are: SMALL GLAM SLAM Pilot 1 : Mediawiki/Wikabase curated lists SMALL GLAM SLAM Pilot 1 : curated lists of ontologies and data models SMALL GLAM SLAM Pilot 1 : Essence modeling with Wikibase experiment Lot’s of fun to come!

- Aryan Kaushik: GNOME Asia Nepal 2023 (2024/02/04 01:27)
  Namaste Everyone! Hi everyone, so it has been a while since the successful completion of GNOME Asia Summit 2023, but well, when you have back-to-back exams, it becomes hard to write up. Last year GNOME Asia happened in Kathmandu Nepal from December 1 - 3. Like GNOME Asia 2022, it was an amazing experience, to say the least. Nepal and India having really close cultural ties made it feel like we are at home without being there, the friendly people, scenic beauty, the superpower to use Hindi in situations where English wasn't recognised, or visiting our holiest sites, Nepal offered us every bit of amazingness we could have asked for. If I get to talk about Nepal, it can be a really big separate blog post, but this is about GNOME Asia, so let's proceed in that regard. Day 0 (Because indexing starts with 0 ;)) Before departing from India, I used my trimmer, you know, to look good, unfortunately, while hearing music I got so lost in it that I didn't secure the adjustable comb properly and trimmed with 0.5mm :D Lost my beard just before the trip, yipeeee. We landed in Nepal on 29th December, and after having some struggle with getting a cab and taking lots of photos, we went on to our hotel. The hotel was really nice, so thanks GNOME for that :D We freshened up a bit and went sightseeing, exploring Thamel, and ate momos (Still like Uttar Pradesh (North Indian state) ones more ;)) We also bought some really needed and mandatory fridge magnets, because if you can't flex it, did it really happen? Day 1 Where do I even start lol, it was probably the most jam-packed day. Waking up in the morning was hard, to say the least, but I conquered it. After freshening up I went for breakfast and had the pleasure of meeting - Rosanna, Kristi, Anisa, Fenris, Jipmang, Matthias and others I met at the previous two conferences. It was nice catching up again. We briefly discussed about next GNOME Asia and UbuCon Asia with stuffed mouths, because that is important as well. Fun fact, we won the bid to host UbuCon Asia 2024 in India and are hoping to have a joint event with GNOME Asia as well. After being the last to finish up (Eating slowly and enjoying fully:P) we proceeded to the venue of the conference. There I met Aaditya Singh (Local team lead). We conversed a few times online and it was great to meet in person. After looting some swags, we went to the conference rooms. Holly (New GNOME ED) started the conference and then we proceeded to the Keynote by Justin, It was just awesome!!!! Probably something I'll recommend my peers to watch. Then we had a talk by Federico and Rosanna, again awesome and then we had the best talk of the conference, mine!! After draining myself with the presentation and the brief Q&A (If the talk felt a bit small and fast-paced, time was the issue) I proceeded to resume my role as an attendee and judging others ;) After attending many more awesome talks, we got to witness the Fedora release party, and even when we were literally on power saving mode, with a 5% charge, the party acted like a charger (Yeah.... not the best at having metaphors) Due to the cake delay, we had a nice intro section, it acted like an icebreaker and made everyone know each other well. After the party I don't know what we did tbh, most probably crashed at the hotel hehe. Phew that was a lot for one day. Fedora and GNOME you are cool ;) Day 2 We had to miss the Keynote on the second day to visit Pashupatinath temple in the morning when it is quiet and peaceful. It is one of our holiest sites and if you visit Kathmandu and don't go there (for Hindus) then you are missing out on the most peaceful place. Those memories still give me chills. But, I did watch the Keynote in Enitrety later on, and I have to say, the only time I watched a video this long it was of Linus Torvalds haha. It was next-level and full of insights. I just wish I could have attended it. I had the pleasure of talking with Hempal sir multiple times, and he was one of the best personalities. Then had the pleasure to listen to amazing talks including by Federico, Nasah, Khairul, and others. Ps from my experience of GUADEC and GNOME Asia, Anisa's would have been great as well. I presented a talk on accessibility at GUADEC, but the one by Federico made me learn so much,

which also makes the point that we need better documentation in this area, as it was painful to find those. We then visited Thamel again, because why not, bought some Pashmina shawls from the hotel, and called it a day! Day 3 Day 3 was a social visit, which means the day when you become poor after buying too much stuff, tired after walking too much, and amazed after exploring the beauty of a place like Nepal. The best part was explaining about our god Shiva and about Shivling to curious Rosanna and Fenris. It was a moment where we shared our cultures and did knowledge exchange. Have to say, I became this spiritual after quite a long time, but it was a nice change. We also ate Newari traditional lunch, where I also tried Tongba, a native alcoholic drink made from fermented millet, boiled milk and herbs, it was the second time I touched Alcohol in my life, and only after getting assurance that the Alcoholic content was too little :) It was some really nice food, and the Tongba was also surprisingly good. There me and Asmit also did some mischief with fire, which was well... childish and fun :) Don't know what happened to me there, but it was probably the most I have ever networked, maybe because after two conferences I became more comfortable and familiar with the community at large. Day 3 was more of an experience so I don't think there is much more to be said for that. Day 4 Visited Monastries and Boudhnath Stupa, and then departed back home. End To end with I want to thank GNOME Foundation for sponsoring my visit and giving me the opportunity to witness the awesome talks in person and bond with the community. Btw, looking forward to meeting many of them again at UbuCon Asia 2024 India and also hopefully GNOME Asia 2024 :D

- Matthias Clasen: GTK hackfest updates (2024/02/03 15:02)
  As we often do, a few members of the GTK team and the wider GNOME community came together for a two-day hackfest before FOSDEM. This year, we were aiming to make progress on the topics of accessibility and input. Here is a quick summary of what we’ve achieved. Accessibility We agreed to merge the socket implementation for webkit accessiblity that Georges wrote We agreed that the accessible notification api that Lukáš suggested is fine We finished the GtkAccessibleText interface and moved our internal implementations over to that interface We discussed the possibility of an a11y backend based on AccessKit Input Carlos reviewed the merge request for passing unhandled events back to the system (on macOS) We looked over the remnants of X11-style timestamps in our input apis and decided to provide alternatives taking an event Wayland Carlos started to turn the private gtk-shell protocol into separate protocols Thanks to the GNOME foundation for supporting this event.
- Daniel García Moreno: Where's my python code? (2024/02/02 23:00)
  Python is a interpreted language, so the python code are just text files with the .py extension. For simple scripts it's really easy to have your files located, but when you starts to use dependencies and different projects with different requirements the thing starts to get more complex. PYTHONPATH The Python interpreter uses a list of paths to try to locate python modules, for example this is what you can get in a modern GNU/Linux distribution by default: Python 3.11.7 (main, Dec 15 2023, 10:49:17) [GCC] on linux Type "help", "copyright", "credits" or "license" for more information. >>> import sys >>> sys.path ['', '/usr/lib64/python311.zip', '/usr/lib64/python3.11', '/usr/lib64/python3.11/lib-dynload', '/usr/lib64/python3.11/site-packages', '/usr/lib64/python3.11/_import_failed', '/usr/lib/python3.11/site-packages'] These are the default paths where the python modules are installed. If you install any python module using your linux packaging tool, the python code will be placed inside the site-packages folder. So system installed python modules can be located in: /usr/lib/python3.11/site-packages for modules that are architecture independent (pure python, all .py files) /usr/lib64/python3.11/site-packages for modules that depends on the arquitecture, that's something that uses low level libraries and needs to build so there are some .so files. pip When you need a new python dependency you can try to install from your GNU/Linux distribution using the default package manager like zypper, dnf or apt, and those python files will be placed in the system paths that you can see above. But distributions doesn't pack all the python modules and even if they do, you can require an specific

version that's different from the one packaged in your favourite distribution, so in python it's common to install dependencies from the Python Package Index (PyPI). Python has a tool to install and manage Python packages that looks for desired python modules in PyPI. You can install new dependencies with pip just like: $ pip install django And that command looks for the django python module in the PyPI, downloads and install it, in your user $HOME/.local/lib/python3.11/site-packages folder if you use --user, or in a global system path like /usr/local/lib or /usr/lib if you run pip as root. But the usage of pip directly in the system is something not recommended today, and even it's disabled in some distributions, like openSUSE Tumbleweed. [danigm@localhost ~] $ pip install django error: externally-managed-environment × This environment is externally managed ╰─> To install Python packages system-wide, try zypper install python311-xyz, where xyz is the package you are trying to install. If you wish to install a non-rpm packaged Python package, create a virtual environment using python3.11 -m venv path/to/venv. Then use path/to/venv/bin/python and path/to/venv/bin/pip. If you wish to install a non-rpm packaged Python application, it may be easiest to use `pipx install xyz`, which will manage a virtual environment for you. Install pipx via `zypper install python311-pipx` . note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can override this, at the risk of breaking your Python installation or OS, by passing --break-system-packages. hint: See PEP 668 for the detailed specification. virtualenvs Following the current recommendation, the correct way of installing third party python modules is to use virtualenvs. The virtualenvs are just specific folders where you install your python modules and some scripts that make's easy to use it in combination with your system libraries so you don't need to modify the PYTHONPATH manually. So if you've a custom project and want to install python modules you can create your own virtualenv and use pip to install dependencies there: [danigm@localhost tmp] $ python3 -m venv myenv [danigm@localhost tmp] $ . ./myenv/bin/activate (myenv) [danigm@localhost tmp] $ pip install django Collecting django ... Successfully installed asgiref-3.7.2 django-5.0.1 sqlparse-0.4.4 So all dependencies are installed in my new virtualenv folder and if I use the python from the virtualenv it's using those paths, so all the modules installed there are usable inside that virtualenv: (myenv) [danigm@localhost tmp] $ ls myenv/lib/python3.11/site-packages/django/ apps contrib db forms __init__.py middleware shortcuts.py templatetags urls views conf core dispatch http __main__.py __pycache__ template test utils (myenv) [danigm@localhost tmp] $ python3 -c "import django; print(django.__version__)" 5.0.1 (myenv) [danigm@localhost tmp] $ deactivate With virtualenvs you can have multiple python projects, with different dependencies, isolated, so you use different dependencies when you activate your desired virtualenv: activate $ . ./myenv/bin/activate deactivate $ deactivate High level tools to handle virtualenvs The venv module is a default Python module and as you can see above, it's really simple to use, but there are some tools that provides some tooling around it, to make it easy for you, so usually you don't need to use venv directly. pipx For final python tools, that you are not going to use as dependencies in your python code, the recommended tool to use is pipx. The tool creates virtualenv automatically and links the binaries so you don't need to worry about anything, just use as a way to install third party python applications and update/uninstall using it. The pipx won't mess your system libraries and each installation will use a different virtualenv, so even tools with incompatible dependencies will work nicely together in the same system. Libraries, for Python developers In the case of Python developers, when you need to manage dependencies for your project, there are a lot of nice high level tools for managing dependencies. PDM hatch micropipenv pip-tools pipenv poetry These tools provides different ways of managing dependencies, but all of them relies in the use of venv, creating the virtualenv in different locations and providing tools to enable/disable and manage dependencies inside those virtualenvs. For example, poetry creates virtualenvs by default inside the .cache folder, in my case I can find all poetry created virtualenvs in: /home/danigm/.cache/pypoetry/virtualenvs/ Most of these tools add other utilities on top of the dependency management. Just for installing python modules easily you can always use default venv and pip modules, but for more complex projects it's worth to investigate high level tools, because it'll make easy to manage your project dependencies and virtualenvs. Conclusion There are a lot of python code inside any modern Linux distribution and if you're a python developer it's possible to have a lot of python code. Make sure to know the source of your modules and do not mix different environments to avoid future headaches. As a final trick, if you don't know where's the actual code of some python module in your running python script, you can always ask: >>> import django >>> django.__file__ '/tmp/myenv/lib64/python3.11/site-packages/django/__init__.py' This could be even more complicated if you start to use containers and different python versions, so keep you dependencies clean and up to date and make sue that you know where is your Python code.

- Jose Hunter: Let's talk about Mingle (2024/02/02 22:54)
  I have spent the past few weeks working on a small personal project, because I do not know when to stop. I think Mingle is mostly feature complete at this point. It lets you play with Google's Emoji Kitchen, which is available on Android's GBoard, on GNOME. It provides a convenient way to find the perfect expression and paste it into any conversation online. It is a GTK4/Libadwaita application written in Vala/Blueprint. I have such a weird fascination with Vala. I have a Java/C# background so the syntax is quite familiar. Blueprint is the future and way more readable then XML. The application is still using a placeholder icon because I am not an artist and my own endeavors into Inkscape have not been too successful. Life has kinda been kicking my butt recently, so working on this app has been a small reprieve from how uncaring and mean existence can be.Personal FavoriteLazy LoadingI started this project under the assumption I would get it running under GNOME Shell Mobile one day. So, I had to implement lazy loading, something I have never done before, if I ever wanted to see this run on a mobile device; Until somewhat recently, when you selected an emoji it would just grab every relevant combination. This was not smart or efficient. Mingle would very quickly use up more memory than my web browser. Google's artists made a lot of emoji art, and loading that many combined emojis and populating a flow box asynchronously at the same time tanked performance. This made my XPS 13, with only 8 gigabytes of memory, scream in agony.Currently, Mingle is loading combined emojis in batches and fetches more on an edge-overshot signal, so it loads more as your scroll. This works both with a scroll-wheel and touch-input (Thanks XPS). I am not sure if this is the best approach, but it prevents the app from being a stuttering mess. When you select a left and a right emoji we prepend that combination to our center flow box.Style SwitcherA fairly common pattern I have seen in both GNOME and Elementary apps are these slick color/style selectors. I wanted to implement this pattern, because why not? It looks good and I get to learn how things work. These custom widgets are really just radio buttons that are heavily stylized with CSS. We then just have to add it as a custom child to our primary menu. Here is great example using JavaScript and Blueprint. Blackbox, my terminal of choice, also has a selector written in Vala and XML. These two projects and the beauty of open source software allowed me to create a solution I am happy with. If anybody is reading this and is interested you can check out the repo. I intend to polish things a bit more and then do a first release on Flathub.
- Jordan Petridis: Thessaloniki spring Hackfests! (2024/02/02 19:03)
  Hello everyone! I am here to terrorize your calendar by dropping the dates for two back to back hackfests we are organizing in the beautiful city of Thessaloniki, Greece (who doesn’t like coming to Greece on work time, right?). May 27-29th we will be hosting the annual GStreamer Spring Hackfest. If multimedia is your thing, you know the drill. Newcomers are also welcome ofc! May 31st-June 5th we will be hosting another edition of the GNOME Rust Hackfest. First in person Rust hackfest ever since the pandemic started. From what I heard, half of Berlin will be coming for this one so we might change its scope to an all around GNOME one, but we will see. You are all welcome! See the pages of each hackfest for more details. We are in the final steps of booking the venue but it will most likely be in the city center and it should be safe to book

accommodation and traveling tickets. Additionally the venue we are looking at can accommodate around 40 people, so please please add yourself to the organizing pad of each hackfest you are interested in, in addition to any dietary restrictions you might have. See you all IRL!

- Christian Hergert: Performance Profiling for Fedora Magazine (2024/02/02 18:58)
  I've authored an article recently for Fedora Magazine on Performance Profiling in Fedora. It covers both the basics on how to get started as well as the nitty-gritty details of how profilers work. I'd love for others to be more informed on that so I'm not the only person maintaining Sysprof. Hopefully I was able to distill the information down a bit better than my typical blog posts. If you felt like those were maybe too difficult to follow, give this one a read.
- Marcus Lundblad: FOSDEMaps (2024/01/31 22:11)
  It's only a couple of days left until that weekend of the week of the turn of month between January and February, and that means it's time again for FOSDEM, gathering FOSS entusiasts in Brussels, Belgium. And I will be there as well!There has been some nice improvements in Maps since the end-of-year post before New Years as well.James Westman has continued improving vector rendering support in libshumate and also implemented the ability to click on symbols, so you can finally click on labels and icons on the map instead of using the „What's here?" context menu item, doing a reverse geocoding. That always felt a bit like a „work around". This seems like one of those cases of „a video says more than a thousand images".This is available when enabling the experimental vector map layer.Another thing I have been working on is improving the experience of the favorites menu. Now instead of showing an insensitive, greye-out button in the headerbar when there are no places marked as favorites now the menu is always accessible, but instead shows an „empty state" when there are no places marked as favorites.When there are places marked as favorites, there is now a close button next to the item allowing to remove it from favorites (rather than having to select the place and animate there to unmark the star icon in the popover showing).When removing a favorite from menu a „toast" is displayed offering to undo this action (similar to when e.g. deleting files in Files).Jakub Steiner has redesigned the map pin icon.Compared to the old Tango-style icon that, while being a nice icon that has served us well looked a bit out-of-place in relation to the new UI style with a more 3D look.Another new feature was thought up when looking for a cafe in a shopping mall in Riga during GUADEC last year is showing information about the floor location of places. There two established tags in OSM for this: level which represents the number of floors relative to the ground floor (or the lowest ground floor for buildings built in a souterrain fascion). In this case we now show this information in a spelled-out form for ground level or above, or below ground level (with provision for using localized plural forms).The other tag is level:ref referring a literal floor designation as „written in the elevator“. This could be fully spelled-out named floors, or numbers with suffixes and so on. When this is available we'll refer to that one as this would directly correspond to actual writing on-site.Lastly James has added support for showing descriptions in the info bubble when clicking on points in a GeoJSON shape files when present. It also now shows the name of the shape file in the info bubble (this shape file was from an old GUADEC in Strasbourg).Maybe I forgot something, but I think those where the highlights of new stuff so far in 2024.Maybe see you in Brussels in a couple of days!
- Bastien Nocera: Re: New responsibilities (2024/01/31 11:33)
  A few months have passed since New Responsibilities was posted, so I thought I would provide an update.Projects MaintenanceOf all the freedesktop projects I created and maintained, only one doesn't have a new maintainer, low-memory-monitor.This daemon is what the GMemoryMonitor GLib API is based on, so it can't be replaced trivially. Efforts seem to be under way to replace it with systemd APIs.As for the other daemons:switcheroo-control got picked up by Jonas Ådahl, one of the mutter maintainers. I'm looking forward to seeing this merge request

fixed so we can have better menu items on dual-GPU systemsiio-sensor-proxy added Dylan Van Assche to its maintenance team, assisting Guido Günther.power-profiles-daemon is now maintained by Marco Trevisan. It recently got support for separate system and CPU power profiles, and display power saving features are in the works.(As an aside, there's posturing towards replacing power-profiles-daemon with tuned in Fedora. I would advise stakeholders to figure out whether having a large Python script in the boot hot path is a good idea, taking a look at bootcharts, and then thinking about whether hardware manufacturers would be able to help with supporting a tool with so many moving parts. Useful for tinkering, not for shipping in a product)Updated responsibilitiesSince mid-August, I've joined the Platform Enablement Team. Right now, I'm helping out with maintenance of the Bluetooth kernel stack in RHEL (and thus CentOS).The goal is to eventually pivot to hardware enablement, which is likely to involve backporting and testing, more so than upstream enablement. This is currently dependent on attending some formal kernel development (and debugging) training sessions which should make it easier to see where my hodge-podge kernel knowledge stands.Blog backlogBefore being moved to a different project, and apart from the usual and very time-consuming bug triage, user support and project maintenance, I also worked on a few new features. I have a few posts planned that will lay that out.

- Sebastian Wick: Booting into Toolbox Containers (2024/01/30 12:23)
  There are a lot of tangible benefits in using toolbox containers for development to the point that I don’t want to use anything else anymore. Even with a bunch of tricks at our disposal, there are still downsides. The containers are not complete sessions but rather try to integrate with the host session. If you’re working on something that is part of a session it might be possible to run a test suite and even more elaborate setups but running it fully integrated often becomes a problem. If the host is a traditional mutable system it’s possible to just not use toolbox. If you’re on an immutable system they often offer some way to make it mutable temporarily using some kind of overlay at which point they behave mostly like the traditional mutable systems. The unfortunate side effect is that you don’t get the benefits of toolbox anymore. It’s also possible to develop in the toolbox and on the host system at the same time, depending on what specifically you’re working on right now to get the benefits of both systems. The drawback is that the toolbox container and the host are different systems. You’re setting up, compiling, etc. everything twice and run your project in different environments. Also not ideal. We can do better. Toolbox can, in theory, use arbitrary OCI images. In practice there are assumptions from toolbox on how an image looks and behaves. Fedora Silverblue, or rather rpm-ostree, can also, in theory, boot arbitrary OCI images but also comes with its assumptions. It turns out that in practice the unofficial OCI image variant of Fedora Silverblue can be used as a toolbox image and the images of such containers can be booted into with rpm-ostree. $ toolbox create -i quay.io/fedora-ostree-desktops/silverblue:39 my-silverblue-toolbox $ toolbox enter my-silverblue-toolbox ⬢ # install dnf to make it behave like a usual toolbox container ⬢ sudo rpm-ostree install -y dnf ⬢ sudo dnf update -y ⬢ # Let's install strace and gdb. Do whatever you want with your container! ⬢ sudo dnf install -y strace gdb ⬢ exit $ # some magic to convert the running container into something rpm-ostree understands $ # there are probably ways to do this with less copying (tell me if you know) $ podman commit my-silverblue-toolbox my-silverblue-toolbox-image $ sudo mkdir -p /var/lib/my-silverblue-toolbox-image $ podman save --format=oci-archive "my-silverblue-toolbox-image" | sudo tar -x -C "/var/lib/my-silverblue-toolbox-image" $ sudo rpm-ostree rebase "ostree-unverified-image:oci:/var/lib/my-silverblue-toolbox-image" $ # boot into our toolbox $ sudo systemctl reboot -i One toolbox container to develop in and one reboot to test the changes in a full session on real hardware. This is all unsupported and might break in interesting ways but it shows the power of OCI based operating systems and toolbox.
- Allan Day: Announcing the GNOME Project Handbook (2024/01/30 11:32)
  I’m a firm believer in the importance of documentation for open source projects, particularly when it comes to onboarding new contributors. To

attract and retain contributors, you need good docs. Those docs aren’t just important for practical information on how to contribute (though that is important). They’re also important when it comes to understanding the more general aspects of a project, like how decisions are made, who the stakeholders are, what the history is, and so on. For new contributors, understanding these aspects of a project is essential to being able to participate. (They are also the aspects of a project that established contributors often take for granted.) Of course, ensuring that you always have up to date project documentation isn’t easy. This is particularly true for large, long-running projects, where the tendency is for large amounts of documentation to get written and then eventually left to rot. As redundant and inaccurate docs accumulate, they increasingly misdirect and impede contributors. This characterization has unfortunately has been true for GNOME for some time. For many years, the main source of project documentation has been the wiki and, for a long time, the vast majority of that wiki content has been either inaccurate or redundant. We can only assume that, with so much out of date information floating around, countless hours of have been lost, with existing contributors struggling to find the information they need, and new potential contributors being put off before they have even gotten started. Enough with the preamble The poor state of GNOME’s documentation has been something that I’ve wanted to tackle for some time, so it’s with great excitement that I’m happy to announce a new, completely rewritten documentation resource for the project: the GNOME Project Handbook (otherwise known as handbook.gnome.org). The handbook is a new website whose goal is to provide accessible, well-maintained documentation about how to get stuff done within GNOME. It has a specific scope: it does not provide technical documentation for those using GNOME technologies, nor does it contain user documentation, nor does it attempt to provide public-facing home pages for apps and libraries. What it does contain is the information required to operate as a GNOME contributor. The fact that handbook.gnome.org is able to have this relatively tight focus is thanks to a collection of other GNOME sites, each of which replaces a role previously played by the wiki. This includes apps.gnome.org, developer.gnome.org, and welcome.gnome.org. Thank you to the creators of those resources! The handbook site itself is managed like any other GNOME project. There’s a repository that generates the site, issues can be reported, and changes can be proposed through merge requests. The hope is that this will avoid many of the maintenance issues that we previously had with the wiki. Notable content The handbook is composed of pages from the wiki, which have largely been rewritten, plus a decent amount of original content. There are some sections which I’m particularly excited about, and want to highlight. Issue tracker guidelines GNOME has had issue reporting and triage guidelines for almost two decades. However, it has been many years since they were actively maintained, and they were never updated when GNOME migrated from Bugzilla to GitLab. I think that a lot of contributors have forgotten that they even exist. The handbook includes a fresh set of issue tracking guidelines, which have been updated for the modern era. They’re based on the old ones from many years ago, but have been substantially revised and expanded. The new guidelines cover how to report an issue, how to review issues for quality and relevance, and policies and best practices for maintainers. One exciting new aspect is guidelines for those who want to get started with issue review as a new contributor. I’m hopeful that having clear processes and guidelines around issue tracking will have an enabling effect for contributors and maintainers, so they can be more forthright when it comes to issue management, and in so doing get our issues trackers into a better state. Governance The handbook has a page on governance! It describes how decisions are made in GNOME, the various roles in the project, who has authority, and how the project works. Us old hands tend to assume this stuff, but for new contributors it’s essential information, and we never documented it before. How to submit a code change Amazingly — incredibly! — until this day, GNOME has not documented how to submit a code change to the project. We just left people to figure it out by themselves. This is something that the handbook covers. If you’ve ever wanted to submit a change to GNOME and haven’t known how, give it a read over. Infrastructure The infrastructure pages aren’t new, but they were previously causing some confusion and so have been substantially rewritten. The new pages aim to make it really clear which services are available, how developer permissions are managed, and how to get access when you need it. What next? It’s still early days for the handbook. Most of the core content is in, but there will be issues and missing pieces. If you spot any problems, there’s an issue tracker. You can also submit merge requests or make suggestions (the project README has more information on this). The plan is to retire the wiki. An exact time line for this has yet to be set (there will be an announcement when that happens). However, it’s encouraged to consult the handbook rather than the wiki from this point forward and, if you’re continuing to use the wiki for anything, to move that content elsewhere. There’s a migration guide with details about how to do this. Many thanks to those who have helped with this project, including but not limited to: Jakub Steiner, Andrea Veri, Emmanuele Bassi, Michael Catanzaro, Brage Fuglseth, Florian Muellner, Alexandre Franke, Sebastian Wick, and Kolja Lampe.

- Tobias Bernard: Save the Date: Berlin Mini GUADEC 2024 (2024/01/29 16:19)
  This year’s GUADEC is going to be in the USA, making it difficult to attend both for Visa/border control reasons and because it’s not easy to get to from Europe without flying. Many of us want to avoid the massive emissions in particular (around 3 tons of CO2, which is half the yearly per-capita emissions in most European countries). If that’s you, too, you’re in luck because we’re doing yet another edition of the fabulous Berlin Mini GUADEC! Berlin has one of the largest GNOME local groups in the world, and is relatively easy to get to from most of Europe by train (should be even easier now thanks to new night train options). At our last Mini GUADEC in 2022 we had people join from all over Europe, including Italy, Belgium, Czech Republic, and the UK. The local Berlin community has grown steadily over the past few years, with regular events and local hackfests such as the Mobile and Local-first hackfests last year, including collaborations with other communities (such as postmarketOS and p2panda). We hope to make this year’s Mini GUADEC another opportunity for friends from outside the project to join and work on cool stuff with us. We’re still in the process of figuring out the venue, but we can already announce that the 2024 Mini GUADEC will cover both the conference and BoF days (July 19-24), so you can already mark it on your calendars and start booking trains :) If you already know you’re going to join, feel free to sign up by adding your name to this Hedgedoc, and join the Matrix room. If you don’t have a GNOME account, please email berlinminiguadec@mailup.net to let us know you’re coming. See you in Berlin!
- Peter Hutterer: New gitlab.freedesktop.org 🚯 emoji-based spamfighting abilities (2024/01/29 07:58)
  This is a follow-up from our Spam-label approach, but this time with MOAR EMOJIS because that's what the world is turning into. Since March 2023 projects could apply the "Spam" label on any new issue and have a magic bot come in and purge the user account plus all issues they've filed, see the earlier post for details. This works quite well and gives every project member the ability to quickly purge spam. Alas, pesky spammers are using other approaches to trick google into indexing their pork [1] (because at this point I think all this crap is just SEO spam anyway). Such as commenting on issues and merge requests. We can't apply labels to comments, so we found a way to work around that: emojis! In GitLab you can add "reactions" to issue/merge request/snippet comments and in recent GitLab versions you can register for a webhook to be notified when that happens. So what we've added to the gitlab.freedesktop.org instance is support for the :do_not_litter: (🚯) emoji [2] - if you set that on an comment the author of said comment will be blocked and the comment content will be removed. After some safety checks of course, so you can't just go around blocking everyone by shotgunning emojis into gitlab. Unlike the "Spam" label this does not currently work recursively so it's best to report the user so admins can purge them properly - ideally before setting the emoji so the abuse report contains the actual spam comment instead of the redacted one. Also note that there is a 30 second grace period to quickly undo the emoji if you happen to set it accidentally. Note that for purging issues, the "Spam" label is still required, the emojis only work for comments. Happy cleanup! [1] or pork-

ish [2] Benjamin wanted to use :poop: but there's a chance that may get used for expressing disagreement with the comment in question

- Matthias Clasen: New renderers for GTK (2024/01/28 17:12)
  Recently, GTK gained not one, but two new renderers: one for GL and one for Vulkan. Since naming is hard, we reused existing names and called them “ngl” and “vulkan”. They are built from the same sources, therefore we also call them “unified” renderers. But what is exciting about them? A single source As mentioned already, the two renderers are built from the same source. It is modeled to follow Vulkan apis, with some abstractions to cover the differences between Vulkan and GL (more specifically, GL 3.3+ and GLES 3.0+). This lets us share much of the infrastructure for walking the scene graph, maintaining transforms and other state, caching textures and glyphs, and will make it easier to keep both renderers up-to-date and on-par. Could this unified approach be extended further, to cover a Metal-based renderer on macOS or a DirectX-based one on Windows? Possibly. The advantage of the Vulkan/GL combination is that they share basically the same shader language (GLSL, with some variations). That isn’t the case for Metal or DirectX. For those platforms, we either need to duplicate the shaders or use a translation tool like SPIRV-Cross. If that is the kind of thing that excites you, help is welcome. Implementation details The old GL renderer uses simple shaders for each rendernode type and frequently resorts to offscreen rendering for more complex content. The unified renderers have (more capable) per-node shaders too, but instead of relying on offscreens, they will also use a complex shader that interprets data from a buffer. In game programming, this approach is known as a ubershader. The unified renderer implementation is less optimized than the old GL renderer, and has been written with a focus on correctness and maintainability. As a consequence, it can handle much more varied rendernode trees correctly. Here is an harmless-looking example: repeat { bounds: 0 0 50 50; child: border { outline: 0 0 4.3 4.3; widths: 1.3; } } gl (left) ngl (right) A close-up view New capabilities We wouldn’t have done all this work, if there wasn’t some tangible benefit. Of course, there’s new features and capabilities. Lets look at some: Antialiasing. A big problem with the old GL renderer is that it will just lose fine details. If something is small enough to fall between the boundaries of a single line of pixels, it will simply disappear. In particular this can affect underlines, such as mnemonics. The unified renderers handle such cases better, by doing antialiasing. This helps not just for preserving fine detail, but also prevents jagged outlines of primitives. Close-up view of GL vs NGL Fractional scaling. Antialiasing is also the basis that lets us handle fractional scales properly. If your 1200 × 800 window is set to be scaled to 125 %, with the unified renderers, we will use a framebuffer of size 1500 × 1000 for it, instead of letting the compositor downscale a 2400 × 1600 image. Much less pixels, and a sharper image. Arbitrary gradients. The old GL renderer handles linear, radial and conic gradients with up to 6 color stops. The unified renders allow an unlimited number of color stops. The new renderers also apply antialiasing to gradients, so sharp edges will have smooth lines. A linear gradient with 64 color stops Dmabufs. As a brief detour from the new renderers, we worked on dmabuf support and graphics offloading last fall. The new renderers support this and extend it to create dmabufs when asked to produce a texture via the render_texture api (currently, just the Vulkan renderer). Any sharp edges? As is often the case, with new capabilities comes the potential for new gotchas. Here are some things to be aware of, as an app developer: No more glshader nodes. Yes, they made for some fancy demos for 4.0, but they are very much tied to the old GL renderer, since they make assumptions about the GLSL api exposed by that renderer. Therefore, the new renderers don’t support them. You have been warned in the docs: If there is a problem, this function returns FALSE and reports an error. You should use this function before relying on the shader for rendering and use a fallback with a simpler shader or without shaders if it fails. Thankfully, many uses of the glshader node are no longer necessary, since GTK has gained new features since 4.0, such as mask nodes and support for straight-alpha textures. Fractional positions. The old GL renderer is rounding things, so you could get away with handing it fractional positions. The new renderers will place things where you tell it. This can sometimes have

unintended consequences, so should be on the lookout and make sure that your positions are where they should be. In particular, look out for out for cairo-style drawing where you place lines at half-pixel positions so they fill out one row of pixels precisely. Driver problems. The new renderers are using graphics drivers in new and different ways, so there is potential for triggering problems on that side. Please file problems you see against GTK even if they look like driver issues, since it is useful for us to get an overview how well (or badly) the new code works with the variety of drivers and hardware out there. But is it faster? No, the new renderers are not faster (yet). The old GL renderer is heavily optimized for speed. It also uses much simpler shaders, and does not do the math that is needed for features such as antialiasing. We want to make the new renderers faster eventually, but the new features and correctness make them very exciting, even before we reach that goal. All of the GPU-based renderers are more than fast enough to render todays GTK apps at 60 or 144 fps. That being said, the Vulkan renderer comes close to matching and surpassing the old GL renderer in some unscientific benchmarks. The new GL renderer is slower for some reason that we have not tracked down yet. New defaults In the just-released 4.13.6 snapshot, we have made the ngl renderer the new default. This is a trial balloon — the renderers need wider testing with different apps too verify that they are ready for production. If significant problems appear, we can revert back to the gl renderer for 4.14. We decided not make the Vulkan renderer the default yet, since it is behind the GL renderers in a few application integration aspects: the webkit GTK4 port works with GL, not with Vulkan, and GtkGLArea and GtkMediaStream currently both produce GL textures that the Vulkan renderer can't directly import. All of these issues will hopefully be addressed in the not-too-distant future, and then we will revisit the default renderer decision. If you are using GTK on very old hardware, you may be better off with the old GL renderer, since it makes fewer demands on the GPU. You can override the renderer selection using the GSK_RENDERER environment variable: GSK_RENDERER=gl Future plans and possibilities The new renderers are a good foundation to implement things that we've wanted to have for a long time, such as Proper color handling (including HDR) Path rendering on the GPU Possibly including glyph rendering Off-the-main-thread rendering Performance (on old and less powerful devices) Some of these will be a focus of our work in the near and medium-term future. Summary The new renderers have some exciting features, with more to come. Please try them out, and let us know what works and what doesn't work for you.

- Hans de Goede: A fully open source stack for MIPI cameras (2024/01/26 16:42)
  Many recent Intel laptops have replaced the standard UVC USB camera module with a raw MIPI camera-sensor connected to the IPU6 found in recent Intel laptop chips.Both the hw interface of the ISP part of the IPU6 as well as the image processing algorithms used are considered a trade secret and so far the only Linux support for the IPU6 relies on an out of tree kernel driver with a proprietary userspace stack on top, which is currently available in rpmfusion.Both Linaro and Red Hat have identified the missing ISP support for various ARM and X86 chips as a problem. Linaro has started a project to add a SoftwareISP component to libcamera to allow these cameras to work without needing proprietary software and Red Hat has joined Linaro in working on this.FOSDEM talkBryan O'Donoghue (Linaro) and I are giving a talk about this at FOSDEM.Fedora COPR repositoryThis work is at a point now where it is ready for wider testing. A Fedora COPR repository with a patched kernel and libcamera is now available for users to test, see the COPR page for install and test instructions.This has been tested on the following devices:Lenovo ThinkPad X1 yoga gen 8 (should work on any ThinkPad with ov2740 sensor)Dell Latitude 9420 (ov01a1s sensor)HP Spectre x360 13.5 (2023 model, hi556 sensor)Description of the stackKernel driver for the camera sensor, for the ov2740 used on current Lenovo designs (excluding MTL) I have landed all necessary kernel changes for this upstream.Kernel support for the CSI receiver part of the IPU6 Intel is working on upstreaming this and has recently posted v3 of their patch series for this upstream and this is under active review.A FOSS Software ISP stack inside libcamera to replace the missing IPU6 ISP (processing-system/psys) support. Work on this is under way. I've recently send out v2 of the patch-series for this.Firefox

pipewire camera support and support for the camera portal to get permission to access the camera. My colleague Jan Grulich has been working on this, see Jan's blogpost. Jan's work has landed in the just released Firefox 122. comments

- GNOME Accessibility Blog: Automated testing of GNOME accessibility features (2024/01/26 14:16)
  GNOME is partipating in the December 2023 – February 2024 round of Outreachy. As part of this project, our interns Dorothy Kabarozi and Tanju Achaleke have extended our end-to-end tests to cover some of GNOME’s accessibility features. End-to-end testing, also known as UI testing, involves simulating user interactions with GNOME’s UI. In this case we’re using a virtual machine which runs GNOME OS, so the tests run on the latest, in-development version of GNOME built from the gnome-build-meta integration repo. The tests send keyboard & mouse events to trigger events in the VM, and use fuzzy screenshot comparisons to assert correct behavior. We use a tool called openQA to develop and run the tests. Some features are easier to test than others. So far we’ve added tests for the following accessibility features: High contrast theme Large text theme Always-visible scrollbars Audio over-amplification (boost volume above 100%) Visual alerts (flash screen when the error ‘bell’ sound plays) Text-to-speech using Speech Dispatcher Magnifier (zoom) On-screen keyboard In this screenshot you can see some of the tests: Here’s a link to the actual test run from the screenshot: https://openqa.gnome.org/tests/3058 These tests run every time the gnome-build-meta integration repo is updated, so we can very quickly detect if a code change in the ‘main’ branch of any GNOME module has unintentionally caused a regression in some accessibility feature. GNOME’s accessibility features are seeing some design and implementation improvements at the moment, thanks to several volunteer contributors, investments from the Sovereign Tech Fund and Igalia, and more. As improvements land, the tests will need updating too. Screenshots can be updated using openQA’s web UI, available at https://openqa.gnome.org, there are instructions available. The tests themselves live in openqa-tests.git and are simple Perl programs using openQA’s testapi. Of course merge requests to extend and improve the tests are very welcome. One important omission from the testsuite today is Orca, the GNOME screen reader. Tanju spent a long time trying to get this to work, and we do have a test that verifies text-to-speech using Speech Dispatcher. Orca itself is more complicated and we’ll need to spend more time to figure out how best to set up end-to-end tests for screen reading. If you have feedback on the tests, we’d love to hear from you over on the GNOME Discourse forum.

From:
https://wiki.tromjaro.alexio.tf/ - **TROMjaro wiki**

Permanent link:
https://wiki.tromjaro.alexio.tf/doku.php?id=news:planet:gnome

Last update: **2021/10/30 11:41**