The Wayland debacle is coming

Volatile Infrastructure is worse than volatile applications

Wayland is a replacement for Xorg. It promises to be better suited for some uses and more secure and faster. And it may be, but it is yet another in a line of breaking infrastructure changes that expect existing programs to adapt to The New Way Of Doing Things™, and the ones that do not follow get semi-broken.

I stayed mostly silent on this for a long time, hoping that people would figure this out, because the ones writing wayland are actually those most competent in display servers, but I now see the same philosophy at work that already caused so much other breakage, so I decided to write.

That philosophy is: someone else has to clean up the mess I made.

With the expectation that if the work for that is not too big, people will do so.

But that's a false assumption, because many useful free software tools are mostly unmaintained. And that is not a weakness: in Free Software we are able to create things that last and build upon the work of those before us, because it continues to last without constant maintenance work. Some small migrations are usually done by distributions who then upstream their changes, but that's it. This is a strength.

Constant breaking changes to the infrastructure threaten that.

> We cannot stand on the shoulders of giants if we constantly break old tools.

I used to think that rewriting our foundations is good. Then I saw the Python 3 debacle.

That was a much smaller change and it still caused a huge amount of unexpected breakage.

Breaking changes are not universally disallowed. You can break things, but if you do, you have to fix them and not just expect others to do that work for you.

I also saw udev (broke how devices work, expected everyone to adapt), pulseaudio (broke Audacity — it hasn't worked well since then and I was without good recording tool until obs arrived, that now gets worse with Wayland), I saw Python 3 (enough said about it), and I saw systemd (broke screen! Yes, there are workarounds and non-default options, but this just broke how I worked on servers, and OpenRC showed that systemd was completely unneeded), and now I see Wayland doing the whole thing again.

All these promise to make the system better, but leave it in a state of eternal semi-brokenness, because before the first breakage is fully resolved, the next infrastructure gets rewritten and it causes more breakage.

That's why I nowadays think that volatile software should never be used — except for experimenting — and that volatile infrastructure is worse.

> Recurrent breaking changes to the infrastructure threaten Free Software as a usable environment to work in.

If we value that some random hacker can create a useful tool that solves a real problem for people, we can't constantly break our infrastructure.

A recent problem is that Wayland breaks accessibility tech (orca and global desktop shortcuts).

The change to Wayland is one that breaks a lot of things, but those who push Wayland do not think it their responsibility to keep all the things working that others have already built to fulfill special needs.

I do not need a11y tech, but I will still see a lot of my tools broken when I am finally forced to switch to Wayland.

If I understand it correctly, thanks to Xwayland apps that don't use Wayland won't be broken. But those using wayland will be inaccessible for a11y.

> The problem of breaking compatibility with tooling is that they expect others to clean up the problems Wayland causes.
> That's a problem in their philosophy.

They aren't the only ones who do that (pulseaudio broke audacity for me, the only sane tool for audio editing), but they are the most prominent right now.

There's the core tenet: do not make your software volatile — Wayland makes other software volatile: after an update, things are broken. And the complex tools are the ones most likely to get broken.

I had hoped that we learned from the Python 3 debacle: it took over a decade for Python 3 to become adopted and even today there are many specialist tools that require Python 2.

> Yes, those tools are not well-maintained, but they work. They solve real problems.

Folks who push Wayland into distributions don't have to keep compatibility with Xorg: they have to take responsibility to fix the breakage caused by required deviation from compatibility.

Here's a reddit-thread of people complaining: https://www.reddit.com/r/linux/comments/13hn54f/the_whole_x11_vs_wayland_thing/

The answers by Wayland fans are dismissive — and that's a problem in philosophy.

If I cause bugs in other areas with a refactoring at work, chances are good that I'll get the bugs assigned. Sometimes the fallout of a refactoring that missed some usecases can be too big for one person, then others will jump in. But in general I cannot just force a change on others and then expect them to clean up after me. That's basic responsibility for the effects of your actions.

And I think that the people pushing Wayland should submit patches to obs and retroarch and all the other tools Wayland adoption breaks to fix the issues. Because they cause them.

Even if you are up for fixing tools, keep in mind that breaking backwards compatibility usually means that the ones who dive deepest into the tools and adapt them to their needs are punished by breakage of their systems. So only break stuff, if there is really no other option. With recurrent breakage we teach people not to tinker, because what you tinkered with will break on update.

> When we break people's tools on an update, we teach them dependence and shallow skills.

One of the big strengths of webbrowsers and one of the big reasons for success of web development which — despite all its brokenness — is taking over most areas of contact people have with computing is that it said: the core rule of web development is "do not break the web". The only reason to break something that exists are real security issues. More precisely: w3: Support Existing Content. And w3c: Evolve rather then revolutionarize.

But why should we care? We're not forced to use Wayland, right? Except we will be, because Wayland is the new kid on the block and new features get implemented there. New features that will at some point be required features for applications. We saw that with python3 and udev and pulseaudio and systemd.

So the responsibility to keep stuff working after an infrastructure change should be with those changing the infrastructure or with those who push these changes on others, not with the developers of applications that use that infrastructure.

> You should pause and carefully consider making a change that will break people's current code. ... Before making a change that's going to cause other people pain, we should ask ourselves if it's really worth the cost. Sometimes

> it is, but many times it's not, and we can wrap the change up so it doesn't hurt anyone. — Volatile Software: A solution

And the amount of unexpected breakage due to infrastructure changes should not be underestimated. Python 3 should have told us that.

List of Links