Developer Switch Dreams
Crafting the GNOME DX

Christian Hergert (christian@hergert.me)

GNOME is so much more than a desktop

What is GNOME?

- Compositor (Mutter, Shell)
- Project with strong design ethic (UX, Hig)
- File Manager (Nautilus, Sushi)
- Hardware Abstractions
  (Bluetooth, Network, Disk, Media, Power, Input)
- Graphics Toolkits (Gtk+, Clutter, St)
- Language Enablement
  (GObject, Python, JavaScript, Vala, Perl, Mono)

What is GNOME? (Continued)

- Content Storage and Exctraction (Tracker, Gom)
- Personal Data Management
  (E-D-S, Calendar, Notes, Tasks, Todo)
- Application Suite (Evolution, Gnumeric, GIMP, Gedit)
- Web Browser (Epiphany, WebKitGtk)
- IPC (D-Bus/kdbus, GVariant, Gsettings)
- Media Framework (GStreamer, pulseaudio, rygel)
- Translations (damned lies, gtranslator)
- Artwork (backgrounds, iconography)

What is GNOME? (Continued)

- Security (PolicyKit, lock screen, libsecret, seahorse, sandboxing)
- Communications (Telepathy, farstream, empathy)
- System Setup/Config (Initial Setup, Control Center)
- File Formats
  (libxml2, documents, json, markdown, ASN.1, desktop)
- Standards
  (wmspec, trash, mime, icons, thumbnail, sounds, recent files)
- Documentation (yelp/mallard, gtk-doc, guides/turorials)

What is GNOME? (Continued)

- Developer Tools
  (glade, gitg, glib, devhelp, gedit, anjuta, Nemiver, Builder, Boxes, ghex, sysprof, terminal, gtk-doc, yelp, d-feet, looking glass, gparted, meld, memprof, vala)

GNOME is a massive achievement

But there is an elephant in the GNU/room

It's really hard to contribute to Free Software

So I did some research on why

Learning how to get started is the largest hurdle in contributing to Free Software

We have hundreds of contributors that would be here right now if we could bridge that gap

Clearly this is unacceptable

We built a fantastic platform

(Now let's help people build things for it)

DX = UX for Developers

DX = UX for Developers

The term “Developer” is very broad

Lots of people, lots of skill-sets, varying expertise

You shouldn't have to be an expert on every part of our platform to begin contributing

We need to provide tools and education to go from beginner, to intermediate, to expert

If more people try our platform,
we'll have more contributors

(It's a numbers game)

So let's make it as easy as possible to get started

Suggestion

Add “Developer Mode” switch to Control Center

Developer Mode Switch

- GNOME SDK runtime made available
- Developer Suite
  (Builder, glade, Nemiver, gitg, code-assistance, toolchain, gedit, devhelp, sysprof)
- “Getting Started for Developers” guides

Suggestion

"GNOME Ambassadors" who can help point new developers in the right direction

(For now you can just join #gnome-love)

Developers need documentation

Developers need Documentation

- It needs to be idiomatic
- It needs to be accurate
- It needs to be at your fingertips when coding

Perfect documentation is improbable

(So allow developers to edit it when it's wrong)

GNOME's design patterns are fantastic

(Now let's make them easy to create)

GNOME's platform libraries are great

(Now let's make them easy to consume)

Consuming platform libraries

- Simplify GObject, GtkWidget, GomResource creation
- Create and consume D-Bus services with ease
- GSettings schema designer
- Demystify GStreamer pipeline creation
- High quality auto-completion
  (Use gir/typelib with dynamic languages)

Developers are always searching for things

(So let's make that fast and comprehensive)

Sometimes the documentation simply doesn't exist

(So provide cross-project search to find code examples)

Releasing applications on distribution life-cycles is no longer a viable strategy

Distribution Life-cycles

- Typically 6-months to 2-years
- Developers often rely on them for new dependencies
  (And no time to build/test against unstable GNOME)
- And now your application broke with the new release
  (Everyone blames each other, ABIs insufficient)
- Application fixed upstream, doesn't get shipped until next release
- Repeat process

Do we ever ship a stable combination?

(Kernel, Drivers, Desktop, Applications)

Even worse, which distribution would you choose to synchronize your releases with?

(Synchronizing with GNOME might be best option)

But what if you want to ship early, often?

Suggestion: xdg-app

xdg-app

- Provide reliable ABI/runtime across upgrades
- Reduce “works for me” bugs via predictable runtime (Development closely resembles production)
- Test application against multiple runtimes (GNOME 3.18, GNOME 3.20, Nightly)
- ISVs can finally ship cross-distribution software
- Parallel installs of “GIMP Beta” finally possible!

Builder should provide an app simulator to make testing multiple GNOME releases painless

(More GNOME Continuous testers means more bugs fixed)

Psst! Stop testing your application in tree!

(If you're interested in testing what your users run)

Builder and xdg-app will help you do this automatically

(Eventually)

Build systems are boring

(Sorta like me)

Build Systems

- Make autotools fast (possible!)
- Make it more declarative (Go Philip Withnall Go!)
- Automate what we can
- Document what we can't automate
- Make alternate build systems possible (cmake, meson)

Developer Outreach

GNOME has the potential to be the
“Desktop for Developers”

Developer Outreach

- Web Development (Django/Rails/Node, HTML/JS/CSS)
- Emerging Languages (golang, rust)
- Internet of Things and Embedded
- Mobile Development
- DevOps/Docker/Atomic

It's my opinion that improving our DX is one of the best things we can do for the effectiveness and longevity of GNOME and Free Software

Questions and Comments?

Please join us at the Builder BoF!

All day Tuesday and Wednesday

Thanks for all your support!

https://www.indiegogo.com/projects/builder-an-ide-of-our-gnome