The Chromium project’s way to Wayland

Automotive Linux Summit
(May/2017)

Antonio Gomes (tonikitoo)
Agenda

● Who is Igalia
● Motivation
● Background
● Developments
Who is Igalia?

- Worker-owned, employee-run Open Source consultancy company, based in Galicia, Spain.
Who is Igalia?

- ~59 employees around the world.
- Areas
  - **Chromium/Blink**, WebKit and Servo;
  - Compilers, JavaScript engines (V8, JSC);
  - Multimedia, Graphics (Mesa), Networking, Accessibility.
Motivation
Motivation

- Not a matter of *trend*, but *timing* instead.
  - Maturity.
  - Demand from different industries.
Motivation

- Being able to run Chromium natively in Wayland-based systems will leverage its adoption in a variety of systems.
  - AGL, GENIVI, Raspberry Pi, Tizen, Bose, Bosch, Volvo, Jolla.
  - **Fedora 25** is shipping Wayland by default.
  - **Ubuntu 17.10** will ship Wayland by default.
  - Major GUI Toolkits have built-in support, including Qt 5, Gtk+, Clutter, EFL.
Background
Background - Ozone/Wayland

- **Ozone/Wayland** (by Intel / 01.org)
  - Aura toolkit: Basic windows, events.
    - ui/views/

- **Ozone** project (original)
  - Abstraction layer for the construction of accelerated surfaces underlying the **Aura toolkit**, as well as input devices assignment and event handling.
  - Backends:
    - DRI -> DRM
      - GBM
      - ChromeOS
    - wayland (off trunk)
      - Linux
Background - Ozone/Wayland

- Browser process
- Renderer process
- GPU process

- desktop integration
- x11
- win
- ozone/wayland
- IPC (old API)
- ozone platform
- wayland connection
- GPU process
Background - Ozone/Wayland

- Good community adoption.

- Project entered in “maintenance mode”
  - December/2015
  - Chromium m49
    - Today’s trunk is m60
In the meanwhile, Ozone layer received two new backends:
  ○ wayland
  ○ x11

Is the problem solved?

no
Background

- May/16 – started experimenting with Chromium’s Ozone/Wayland.
  - Ported part of the code from 01.org to Chromium ToT.
Background

- content_shell ozone/wayland
Background

- Igalia got in touch with Google/Chromium developers to understand the plans for //ui/ozone/platforms/wayland.
  - “servicification”
  - figured ChromeOS plans for mus+ash.
    - Ash
    - Mus (//services/ui/)
Background

- **Ozone** project
  - Abstraction layer for the construction of accelerated surfaces underlying the **UI Service** (aka Mus), as well as input devices assignment and event handling.
  - Backends:
    - ChromeOS
      - DRM / GBM
      - wayland
      - x11
    - Linux
Background

- The original “desktop integration” approach taken in Ozone/Wayland did not comply with the way future Linux desktop Chrome was foreseen.
New developments

Phase 1 - The bring up
Phase 1 - status check

- Prior to Sept/16
  - Chromium ToT had a Wayland backend of Ozone.
    - Partial upstream.
    - still behind in terms of functionality if compared against Intel’s implementation.
  - ChromeOS / mus+ash oriented.
  - Outdated Ozone documentation.
  - Limited buildbot coverage.
Phase 1 - the bring up

- Sept-Oct/16
  - Igalia brought up of Ozone’s Wayland backend in ToT.
  - Start experimenting with “Ozone != ChromeOS”.
- Documentation
- Buildbots

- Design discussions with Robert Kroeger (Google).
Phase 1 - demo

- Nov-Dec/16
  - CES demo: Linux/AGL/Wayland on R-Car M3
Phase 1 - Desktop integration

Linux desktop integration (01.org)

Mus Linux desktop integration

Browser process

Renderer process

Gpu service (thread)

UI process

Window Server

ozone / wayland (connection)

ozone / x11

Gpu service (thread)
New developments

Phase 2 - Chrome / Mus
Phase 2 - ChromeOS

- CrOS has a Window Manager (WM) and a ScreenManager (SM).
- *acceleratedWidget* paired with a display (physical).
  - SM changes *acceleratedWidgets* (e.g. display resolution change).

- **Internal-window mode**
  - All the aura windows in the system end up sharing a single display.
  - Chrome aura and other app windows are embedded within a single top-level *acceleratedWidget*.
Phase 2 - Desktop Chrome

- Desktop Chrome has no SM.
- Desktop Chrome has no WM.
  - One `acceleratedWidget` per Chrome window.
  - User manipulates `acceleratedWidgets` via the host OS window.
    - maximize, minimize, dragging.

- **External-window mode**
  - Modify Chrome and Mus so that Mus creates native `acceleratedWidget`'s for each top-level mus window.
  - chrome/mus.
Phase 2 - Dilemma

- Cut the assumptions that there is SM and WM?
- Fake a WM? Chrome is the WM?
Idea 1: Extending internal win mode

- Original plan proposed for ‘external window’ mode:
  - Create a new “desktop-stub” replacement for Ash?
  - Desktop integration.
    - In essence, a subset of functionality currently provided by Ash is delegated to the native window system.
  - Considering using //src/mash/simple_wm as starting point?

After talking to rjkroewege@, sky@, we agreed that this is not the best way to approach to tackle the issue. Alternatively, sky@ proposed to work this out directly on LinuxOS/Ozone builds.
Idea 2: Mus’ External Window Mode

- Extend Mus and Ozone to support ‘External Window’ mode.
  - Generalize ChromeOS assumptions.

- No major functionality loss if compared to stock Chromium.
Mus’ External Window Mode - breakdown

- Extend the `mus_demo` to work in ‘external window’ mode.
- Rework internal window mode assumptions in the code
  - 1:1 relation of `ws::Display` and `display::Display`.
- Extend Mus to support ‘external window mode’.
- Extend Ozone to work on ‘external window’ mode.
- Make the code that handles the existing `-mus` command line parameter non-ChromeOS specific.
  - Chrome today launches the same way it ought to, for Chrome/Mus.
Mus’ External Window Mode - Status

- What is the status today?
  - Very functional and promising, but WIP.
Mus’ External Window Mode - The project

- The project is being hosted on [GitHub](https://github).
  - well defined contribution policy:
    - peer review.
    - Build bot running existing tests
      - `mus_demo_unittests` (extended to launch multiple windows).
      - `mus_ws_unittests`.
  - Rebase strategy
    - Igalia’s excellence in carrying forks of downstream project forward.
    - Weekly rebases.
    - Continuous history clean up.
  - Periodic sync up with Google.
Mus’ External Window Mode - Status

● Today (Chromium ToT):
  ○ Ozone implies ChromeOS.
    ■ mus+ash == ChromeOS

● Today (GitHub):
  ○ Ozone runs on both ChromeOS and LinuxOS
    ■ mash (ChromeOS + internal window mode).
    ■ mus (LinuxOS + external window mode).
Mus’ External Window Mode - TODO

- Continue to fix the windowing integration when Chromium’s builtin window decorations are used.
  - window resize and dragging.
- Fix drag and drop.
- Fix clipboard (it works as in internal window mode).
- Support newer shell protocols
  - e.g.: XDG v6, supported by Fedora 25.
- Provide updated yocto builds on Igalia’s meta-browser fork.
Mus’ External Window Mode - TODO

- Ensure no feature losses when compared to stock Chromium X11/Linux.
- Ensure there is no performance penalties when compared to stock Chromium X11/Linux.
- Start to upstream some of the changes.

*igalia*
Discussion: UI / GPU split

- chrome --mash (and --mus) still runs the UI and GPU components in the same process but separate threads.
  - Future: musws and musgpu in separate processes
    - https://crbug.com/643746
  - owner: rjkroege@

- Mojo-fication of Ozone/Wayland
  - Use approach similar to Ozone DRM/GBM (ChromeOS)?
  - GBM surface
    - rjkroege: to be discussed later.
Questions?

tonikitoo@igalia.com - Antonio Gomes

mscho@igalia.com - Mi Sun Silvia Cho