The Chromium/Wayland project

WebEngines Hackfest
(Oct/2017)

Antonio Gomes, tonikitoo@
Maksim Sisov, msisov@
Agenda

- About Igalia
- Goals & Motivation
- Background
- Developments
- Demonstration
About Igalia

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

- ~62 employees around the world.
- Areas
  - Chromium/Blink, WebKit and Servo;
  - Compilers, JavaScript engines (V8, JSC);
  - Multimedia, Kernel, Networking;
  - Accessibility, Virtualization & Cloud.
About Igalia
Goals & Motivation
Goal

- Being able to run Chromium natively on Wayland-based systems.
Motivation

● Wayland is a mature solution.
● Demand from different industries.
  − Automotive
  − Mobile
  − Desktop
Background
Background - Ozone/Wayland

- By Intel / 01.org.

- **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

Desktop integration

- Browser process
  - desktop integration
  - x11
  - win
- Renderer process
- GPU process

Desktop integration (01.org)

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

- Good community adoption.
- **Project entered in “maintenance mode”**.
  - December/2015.
  - Chromium m49.
    - Today’s ToT is m63.
Background - Cr Upstream (1/)

- In the meanwhile, Ozone layer in ToT received two new backends:
  - x11
  - wayland

- Is the problem solved? no

- The original “desktop integration” approach taken in **Ozone/Wayland** did not comply with the way future Linux desktop Chrome is foreseen.
Background - Desktop integration

Linux desktop integration (01.org)

- Browser process
  - desktop integration
  - x11
  - win
  - ozone/wayland
- IPC (old API)
- GPU process

Mus Linux desktop integration

- Browser process
  - desktop integration
  - x11
  - win
  - aura/mus
- IPC (Mojo API)
- Window Server
  - ozone / wayland (connection)
  - ozone / x11
- Gpu service (thread)
- UI Service
Background - Cr Upstream (2/)

- **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
      - x11
      - Wayland
    - Linux


New developments

Phase 1 - The bring up
Phase 1 - The bring up

- **Sept-Oct/16**
  - Igalia brought up of Ozone’s Wayland backend in ToT.
  - Experimented with “Ozone != ChromeOS”.
  - [Documentation](#)
  - [Buildbots](#)
Phase 1 - CrOS

- **Internal-window mode**
  - CrOS has a Window Manager (WM) and a ScreenManager (SM).
  - Chrome and other app windows in the system
    - end up sharing a single display.
    - are embedded within a single top-level `acceleratedWidget`. 


Phase 1 - Desktop Chrome

- **External-window mode**
  - Desktop Chrome has no WM.
    - One *accelerated.Widget* per Chrome window.
    - User manipulates *accelerated Widgets* via the host OS window.
      - maximize, minimize, resizing, dragging, fullscreen.
  - Desktop Chrome has no SM.
Phase 1 - Demo

- Nov-Dec/16
  - CES demo: Linux/AGL/Wayland on R-Car M3.
  - meta-browser
Phase 1 - Perf

- Nov-Dec/16
  - Performance on BrowserBench GPU tests
New developments

Phase 2 - Chrome / Mus
Mus’ External Window Mode (1/)

- Modify *IWM* so that it creates native *acceleratedWidget*’s for each top-level window.
  - **Extend Mus and Ozone** to support ‘External Window’ mode.

- No major functionality loss if compared to stock Chrome.
Mus’ External Window Mode (2/)

- 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 (3/)

- Added support to:
  - XDG v6.
  - Keyboard events.
  - Mouse cursors.
  - Touch events (thanks to Collabora!).
  - Multiple windows.
  - Built-in window decoration.
  - Window closing.
  - Menus and widgets.
- Support to common windowing features:
  - maximize, minimize, restore, fullscreen, dragging and resizing.
Mus’ External Window Mode (4/)

- Changed ownership model of some objects.
- Implemented keyboard/IME service integration.
- Implemented a slightly custom “window tree hierarchy”.
- Reworked our “access policy”.
- Followed mushrome’s process model.
- Worked extensively on stability and hardness of our impl.
Mus’ External Window Mode (5/)

- What is the status today?
  Ready for alpha testing.
Mus’ External Window Mode (6/)

- Performance improvements (½)
Mus’ External Window Mode (7/)

- Performance improvements (2/2)
May/17

Performance on BrowserBench GPU tests
About the project (1/)

- The project is being hosted on GitHub.
- Well defined contribution policy:
  - Peer review.
  - Buildbot running existing tests:
    - `services_unittests` and `ozone_unittests`.
    - `mus_demo_unittests` (extended to launch multiple windows).
About the project (2/)

● Rebase strategy:
  ○ Weekly based.
  ○ Continuous history clean up.
    ■ git commit --fixup <SHA>
    ■ Eliminate commit + revert “commit” pairs.
    ■ Use of [DoNotCarryForward] tag.

● Periodic sync up with Google.
TODO

- Fix drag and drop.
- Fix clipboard.
  - it works as in internal window mode.
- Multi screen support.
- Non-english keyboard layouts.
- Ensure no feature losses or major performance penalties when compared to stock Chromium X11/Linux.
- Start to upstream the changes.
TODO

- Integration with AGL.
- Release desktop installers (.deb .rpm).
Breakout session

- Upstream strategy
  - walkthrough of our impl.
  - wayland security review.
- UI / GPU split
  - Future: musws and musgpu in separate processes.
  - [https://crbug.com/643746](https://crbug.com/643746)
- Mojo-fication of Ozone/Wayland.
Questions?

tonikitoo@igalia.com - Antonio Gomes

msisov@igalia.com - Maksim Sisov