Multimedia in WPE
Current status & plans

Philippe Normand

philn@igalia.com

GStreamer conference

Edinburgh

25th-26th October 2018

igalia

Who am I?

- Fiddling with WebKit and GStreamer since 2009
- WebKit committer and reviewer
- GStreamer committer
- Partner at Igalia
  - Worker-owned coop, currently around 70 happy Igalians around the world
  - Provides consulting services for various Free Software projects

Talk Outline

- What is WPE
- Basic infrastructure for Media playback
- Adaptive streaming in HTML5: Media Source Extensions
- WebRTC
- Media-capabilities
- WPE in the wild: i-MX6

WPE, Web Platform for Embedded

WPE, general architecture

- Relies on WebKit's multi-process (mainly WebProcess, NetworkProcess and UIProcess)
- HTML RenderTree layer composition in WebProcess
- Final presentation of composited image deferred to out-of-tree backends
- WPEBackend selected at runtime by the UIProcess (Browser)
- Stripped down GLib public API

libwpe

- Previously known as WPEBackend
- Dependency on libxkbcommon for keymapping
- ViewBackend for rendering
- EGL renderer backend

WPEBackend-fdo

- Relies on wayland-egl
- Cross-process buffer sharing
- API for:
  - EGLImages
  - Or wl_resource objects
  - Or Linux dma-buf information (already used internally)
- Combined with Mesa
- Works on desktop & embedded

Infrastructure for <video>, <audio> & WebAudio

<audio> & <video> in WPE

- Playbin-based MediaPlayerPrivate implementation
- Playbin3 support! 
  - Streams collection handling
  - Good match in-band tracks support!
- GL Video rendering with a custom appsink
- Custom GstAllocator using WebKit's FastMalloc 
- White list of supported containers and codecs
  - Codec installer support effectively useless
- AV1 decoding support! 

igalia

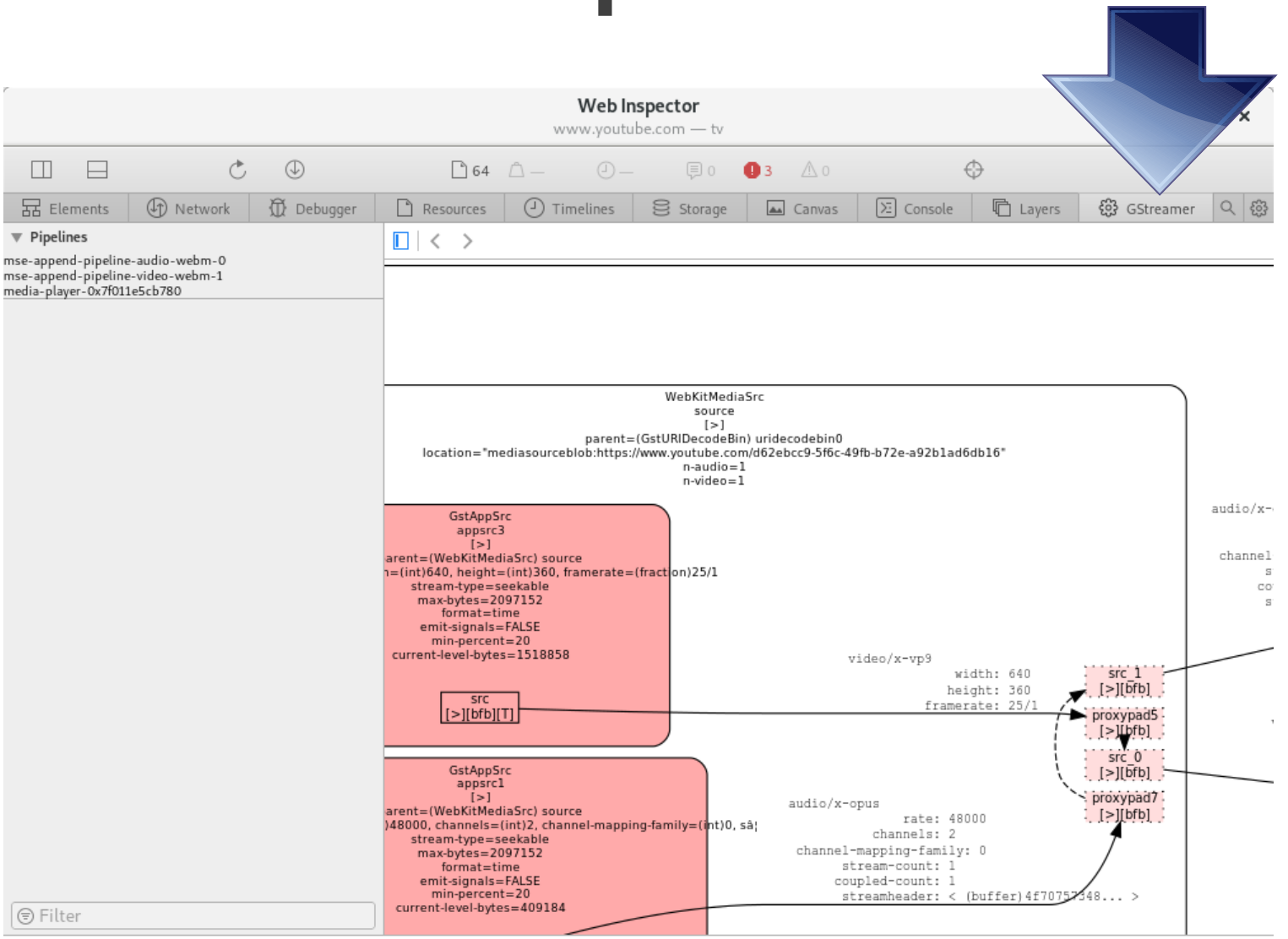
WebAudio

- Current backend is ~ stable
- Decoding pipeline still relying on decodebin
- Playback pipeline
  - Currently using a custom bin containing appsrc elements → interleave
  - Soon:
    - Leverage planar audio support?
    - Rewrite source element based on audiosrc

Debug tooling

- Pipeline dumps?
  - Who likes GST_DEBUG_DUMP_DOT_DIR ?
- Per-pipeline debug logs?
  - GST_DEBUG=wat¿?
- Gst-debugger?
- Tracers?
- ...

Gst Web-Inspector: Soon!



igalia

Adaptive streaming: MSE

By Alicia Boya & Enrique Ocaña

The MSE backend, TL;DR

- Chunks queued from JavaScript world to a SourceBuffer
- One GStreamer WebKit Append pipeline per SourceBuffer
  - Demuxing and parsing of samples
  - Samples stored at WebCore's MSE layer
- Playback pipeline using a dedicated MediaPlayerPrivate implementation
  - Playbin-based
  - Custom source bin element (one appsrc per SourceBuffer)

igalia

MSE-related improvements in GStreamer

- Quite a few improvements in qtdemux for:
  - Samples demuxing in push-mode
  - Edit list support for push-mode
  - Segment event handling
  - Duration-related bug fixes
  - => Around 15 patches so far!
- Matroskademux improvements
  - Emit no-more-pads earlier (after parsing Tracks) (used to be sent while processing the first Cluster)
  - Multi-Tracks parsing
  - Fixes for WebM byte-stream format handling

Current status & plans

- MSE enabled in GNOME-Web!
- MSE backend widely tested on embedded platforms (RPi, i-MX6, …)
- Infrastructure available for combination with EME
- Youtube (“desktop” and */tv) relying on MSE*
  - *VP9 & opus*
  - *H.264 & AAC also supported*
- *Playbin3 / Stream-collections support: planned*
- *Multi-track SourceBuffer support: planned*

WebRTC

By Thibault Saunier & Alex G. Castro

WebRTC Musical chairs

- 2015-2016: OpenWebRTC backend
- 2016: Apple open-sources their LibWebRTC backend
- 2017: OpenWebRTC fades away, backend removed from WebKit trunk
- 2018: WPE and WebKitGTK adopt LibWebRTC with GStreamer platform support

Why LibWebRTC

- Mature and stable (not the API though!)
- Feature complete
- Very active development team
- Existing infrastructure in WebKit

LibWebRTC + GStreamer

- Leverage GStreamer's hardware integration support
  - GstDeviceMonitor
  - Encoders via encodebin
  - Decoders via decodebin
  - Communication with LibWebRTC using appsrc & appsink
- <video> playback integration with a custom src element

igalia

The future, webrtcbin?

- Licensing issues related with boringssl in libwebrtc (for GPL WebKit apps like Epiphany)
- Webrtcbin would be a perfect fit for WebKit!
- Experimental WebKit webrtcbin backend written in November 2017

  https://github.com/philn/webkit/tree/gstwebrtc

igalia

Media-Capabilities

The (draft) spec

- https://wicg.github.io/media-capabilities/
- Goal: provide hints to WebApps regarding the most optimal media encoders & decoders
  - Input: description of the media format (contentType, width, height, framerate, …)
  - Output: 3 booleans:
    - supported
    - smooth
    - powerEfficient

GStreamer “probing”

- New “Hardware” element metadata Classifier (=> 1.16) 
- Elements may implement probing for their NULL→READY state transition 
- Possibly refine Caps templates to reflect what the hardware supports 

- WebKit GStreamer MediaCapabilities backend started

WPE/GStreamer on i-MX6 QuadPlus with Yocto

Yocto layers

- https://github.com/Igalia/meta-webkit/
  - WPEBackends
  - Cog browser!
- https://github.com/OSSystems/meta-gstreamer1.0
  - Updated GStreamer 1.14.x recipes
- (meta-freescale)

Option 1 : Proprietary Freescale driver

- Usable WPEBackends:
  - WPEBackend-RDK/wayland (usable in Weston)
  - WPEBackend-RDK/viv-imx6 (usable in framebuffer)
  - WPEBackend-fdo (usable in Weston)
- GStreamer-imx plugins

Option 2 : Open-source etnaviv driver

- Requires very recent kernel (4.19), Mesa (18.2.2), Wayland (1.16), GStreamer (1.14.4)
- Usable WPEBackends, only working in Weston:
  - WPEBackend-RDK/wayland
  - WPEBackend-fdo (recommended)
- Upstream v4l2 plugin from gst-plugins-good for hardware decoding (and encoding) support


igalia