```
                static void
 erties(GObjectClass
    *gobject_class)
                  {
         ec *pspec;
```

WPE

Current Status & Future

Žan Doberšek

zdobersek@igalia.com

igalia

WPE – What It Is

What It Is

- WebKitGTK+’s younger brother
  - Primarily sheds the GTK+ dependency
  - Not bound to any toolkit or platform
- Vanilla Web
  - Adjustable, low-level Web content embedding

What It Is

- Work started in 2014
  - Work presented at the 2014 Hackfest
  - Another update at the 2016 Hackfest
  - I guess we’re on a 2-year schedule now
- Upstreamed to webkit.org in April 2017

Adaptability

- WPE port of WebKit works against interface definitions
  - “Render targets” for composition of Web content
  - “View backends” for device input and visual output management
- Up to deployers to provide platform-specific interface implementations (in a separate, runtime-loadable library)
  - Enables running on a variety of hardware platforms
  - Low-level approach not imposing many limits

WPE – Current Status

Releases

- Stable releases aligned with WebKitGTK+
  - 2-for-1 branch management
  - 2.22 is the current release
- GObject-based API
  - Stabilized
  - Again something shared with WebKitGTK+

The Interface Library

- Started as “libWPEBackend”
    - Renamed to **libwpe**
- Stabilized API
    - Still could change in the long-term
- Picked up the libxkbcommon dependency
    - Necessary for common keymapping functionality

Reference Backend Impl Library

- **libWPEBackend-fdo**
- Internally uses wayland-egl capabilities
  - “cross-process buffer sharing”
- Provides graphics buffer resource exporting APIs
  - EGLImages
  - wl_resource objects
  - Linux dma-buf information data (soon)
- fdo – freedesktop.org (Mesa)

Testing The Thing

- MiniBrowser
  - Simple Web view app, kept inside the WebKit tree
  - Works as a Wayland client
- Cog
  - Reference testing browser
  - Can be powered by either GTK+ or WPE port
- ~~Dyz~~
  - Too much Lua

Where It’s Used

- … that we know of

- Set-Top Boxes
- Home appliances, Entertainment devices
- In-flight, In-vehicle infotainment systems
- Digital signage

WPE – What’s To Do

Disclaimer

- A lot of this has already been in the works
  - With big improvements
- Scope or repetitiveness of these items simply results in repeated or long-term presence on such lists
- All items apply to WebKitGTK+ as well

Multimedia

- MSE, EME
  - Tracking yearly certification suites
- WebRTC
  - OpenWebRTC deserted
  - Praise the libwebrtc overlord

Graphics

- Threaded Cairo painting
  - Relative benefits
- Get the GPU involved
  - Offload painting to that
  - Fonts will be fun
  - Following the trailblazers
- Vulkan by 2020?
- GPU process

Graphics

- WebGL2
  - Work started, then stalled
  - Continued by Apple
- WebGPU
  - Still in the spec phase, no capacity to participate there
  - Would have to be Vulkan-based
  - ANGLE?

Network & Security

- Libsoup coup
  - (We’re the maintainers now)
- HSTS
- Sandboxing
  - Via Flatpak
  - Or DIY on capable platforms

Standards

- EME, WebPackage, ImageBitmap, WebDriver
- Web Predictability
- Easy to adopt commonly-implemented standards
  - <3 WebKit
- Web Platform Tests integration
  - Integrate it into the QA process

JavaScriptCore

- 32-bit JIT maintenance
  - Alas, we need it
- BigInt, class fields

WPE – Weekend Projects

Different Realities

- What do to with AR/VR?
- OpenVR – existing content, but otherwise abandoned
- OpenXR – in spec phase, prototypes available
- Or start embedding Web content into XR (XR browsers)

Android

- Shoddy backend implementation somewhere on my disk
- Still needs a lot of glue on top to integrate into the process model
- Just a prototype – far from being an usable browser or runtime

Questions?

igalia