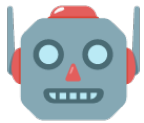
WPE Android  State of the Bot

WebKit Contributors Meeting · 2025

Adrián Pérez de Castro
aperez@igalia.com

igalia

About Me

- CS Engineer, partner of Igalia.
- Systems person turned web engine developer.
- WebKit jack of all trades since ~2012.
- Current focus: platform layer, hardware bringup, release engineering.
- I like old computers, too!

Agenda

1. What, why?
2. When, what?
3. Now, what?

What, Why?

What?

Not a WebKit port.

What?

Not a WebKit port.

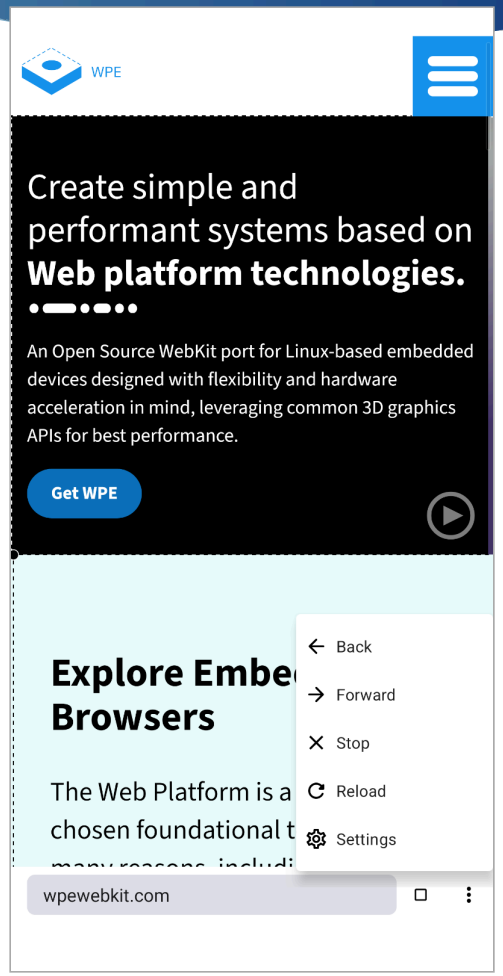
WPE WebKit

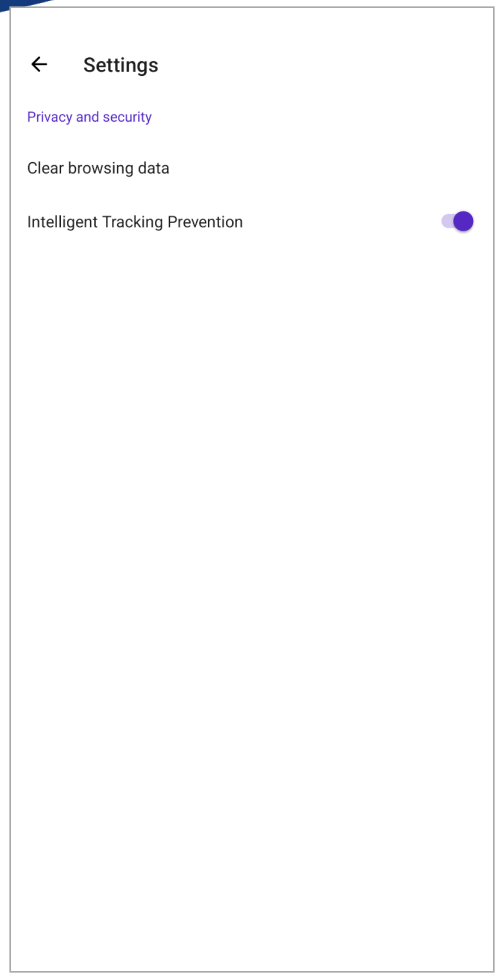
+

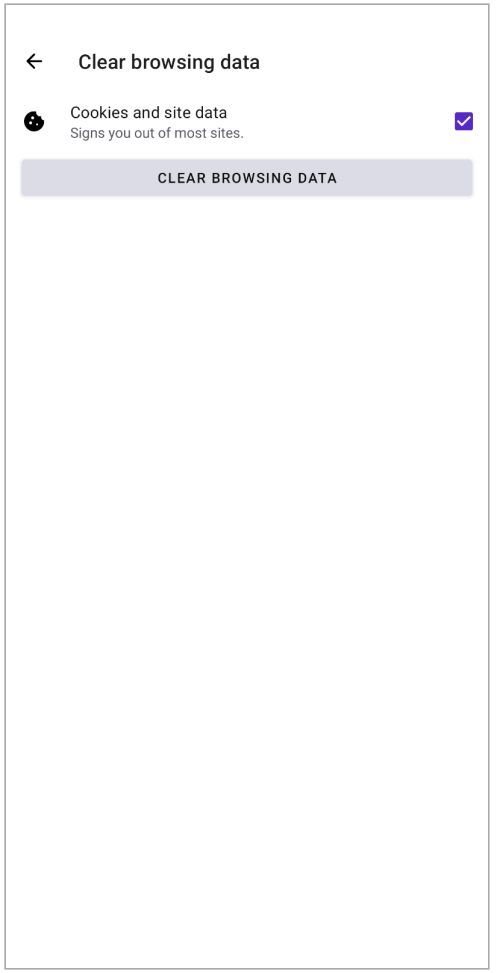
Android **backend**

+

Custom **widget**.







What, Why?

Adrián Pérez de Castro, 2025-10-28

Motivation

Android as **base system**:

- Support by hardware manufacturers (BSPs).
- Optionally without ART/JVM and frameworks.
- Stable target platform.

WPE WebKit exists:

- Bring back WebKit to Android systems.
- Reuse it, no need for a full-fledged WebKit port.
- Prove its adaptability.

More Motivation

Provide a Web engine option that is more customizable.

Avoid reliance on Chromium / Google.

More Motivation

Provide a Web engine option that is more customizable.

Avoid reliance on Chromium / Google.

Contribute to a more **diverse Web engine ecosystem**.

When, what?

Project History

2017 Project starts, basic demo.

2021 WebView widget API based on `GLSurfaceView`.

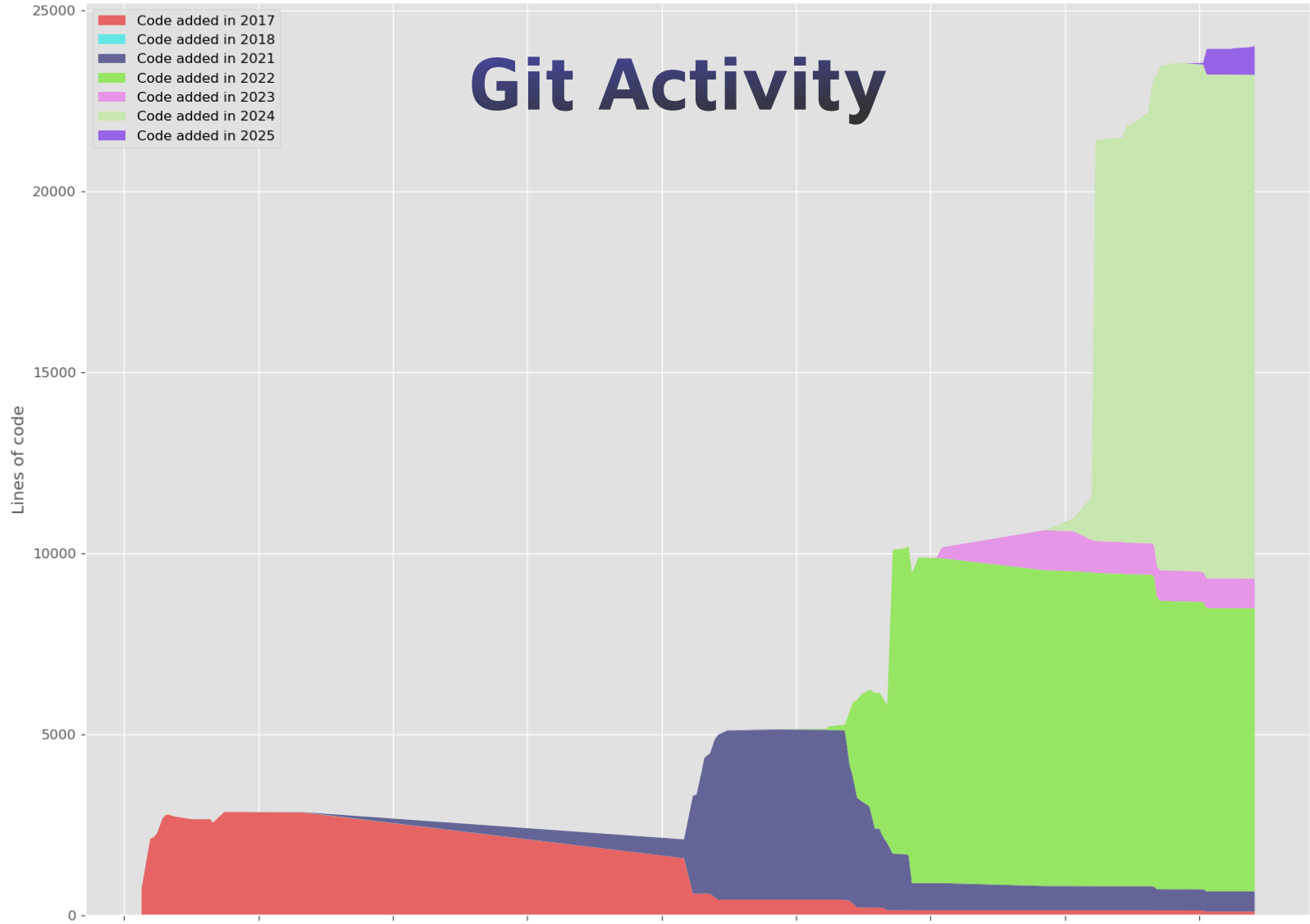
2023 Development picks up again, NLnet grant.

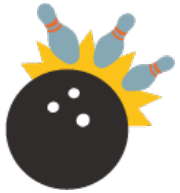
2024, March Drop 32-bit, mainloop integration, PSON, multimedia hardware decoding, WebGL, WebDriver.

2024, August Many patches upstreamed, HTTP/2, output scaling, JS dialogs, RWI, more WebView API.

2025, June Skia, WebKit builds OOTB, Android 15, WebKit 2.50, Maven repository.

2025, October `AHardwareBuffer`, WebXR.


Git Activity
Code added in 2017
Code added in 2018
Code added in 2021
Code added in 2022
Code added in 2023
Code added in 2024
Code added in 2025
Lines of code
25000
20000
15000
10000
5000
0
2017
2018
2019
2020
2021
2022
2023
2024
2025



Done: Preparation

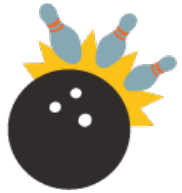
Introduced `OS(ANDROID)`

👉 293096

When, what?

Adrián Pérez de Castro, 2025-10-28

Done: Logging

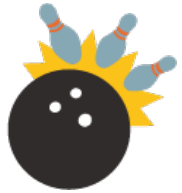


Debugging on Android = 😦

Rebuilding ad-hoc with logging enabled = 😒

Logging `stderr` on Android in native code = 😭

Done: Logging



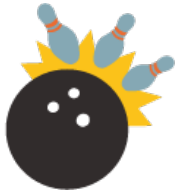
Debugging on Android = 😩

Rebuilding ad-hoc with logging enabled = 😒

Logging `stderr` on Android in native code = 😭

```
adb shell setprop debug.log.WPEWebKit 'WebGL,Media=error'
adb shell setprop log.tag.WPEWebKit VERBOSE
adb logcat -s WPEWebKit
```

👉 297467 + 297472 + 297760 + 299191 + Cleanups

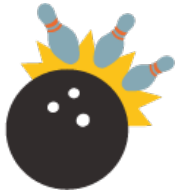


Done: Shared Memory

POSIX shared memory on Android = 🙅

`/dev/shm` on Android = 🙅

`/dev/ashmem` = 😕



Done: Shared Memory

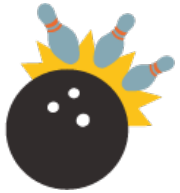
POSIX shared memory on Android = 🙅

`/dev/shm` on Android = 🙅

`/dev/ashmem` = 😕

Use the ASharedMemory API.

👉 293579

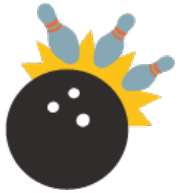


Done: Graphics Buffers

GBM on Android = 🙅

libdrm on Android = 🙅

`/dev/dri` on Android = 🙅



Done: Graphics Buffers

GBM on Android = 🙅

libdrm on Android = 🙅

`/dev/dri` on Android = 🙅

Use the AHardwareBuffer API.

👉 298619 + 299311 + 300466 + Cleanups

Aside: Cleanups

Assumptions about GBM and/or libdrm usage.
Many symbols with `DMABuf` in their name.

Aside: Cleanups

Assumptions about GBM and/or libdrm usage.
Many symbols with `DMABuf` in their name.

Renaming, refactoring.
Easier to integrate more buffer types now.

👉 297914 + 297929 + 298029 + 298034 + 298497 + 299614 + More

Cooking: WebXR

Supported via OpenXR, both in WPE and GTK.

Rewritten to match the Cocoa port architecture.

`AHardwareBuffer` on Android; legacy API + WPEPlatform.

Cooking: WebXR

Supported via OpenXR, both in WPE and GTK.

Rewritten to match the Cocoa port architecture.

`AHardwareBuffer` on Android; legacy API + WPEPlatform.

👉 More in Sergio's talk.

Cooking: AHardwareBuffer

AcceleratedSurface/TextureMapper (PR #48387).

Video buffers (no ETA).

Cooking: Page Icons, WebExtensions

No `WebKitFaviconDatabase` in the WPE port = 🤨.

When, what?

Adrián Pérez de Castro, 2025-10-28

Cooking: Page Icons, WebExtensions

No `WebKitFaviconDatabase` in the WPE port = 🤔.

Improved page icons API (bug #301086).
WebExtension support WIP for GTK and WPE (no ETA).
Side quest: `WebKitImage`.

Now, What?

Behold: The Future

- Track the WebKit `main` branch.
- Switch to the WPEPlatform API.
- Add more WebView APIs.
- Quality-of-Life improvements.
- Continue regular maintenance.
- More system integration: `PowerManager`, long-tap gesture, geolocalization, etc.
- Reduce binary sizes.
- Improve (Mini?) Browser.

Problem: API Verboseness

Cog Launcher

~1.4K LoC

~3.5K (Wayland)

~4.7K (backend)

MiniBrowser

~600 LoC ~1K (backend)

WPEPlatform

32 LoC.

WPEPlatform: Example

```c
#include <wpe/webkit.h>

int main(int argc, const char *argv[]) {
    g_autoptr(GMainLoop) loop = g_main_loop_new(NULL, FALSE);
    g_autoptr(WebKitWebView) view = webkit_web_view_new(NULL);

    webkit_web_view_load_uri(view,
        (argc > 1) ? argv[1] : "https://wpewebkit.org");
    g_main_loop_run(loop);
    return EXIT_SUCCESS;
}
```


Problem: API Loose-ness

Classic

- Ad-hoc, plain C.
- Very flexible.
- Naming is hard.
- Confusing what does what.
- Does not nudge in the right direction.

WPEPlatform

- Based on `GObject`.
- UIProcess-only.
- Better documentation.
- Encourages good practices.

WPEPlatform

Bonuses

- More opportunities for graphics buffers zero-copy.
- Better render pacing, `DisplayLink`.

Cons

- No process launching extensibility (yet).
- API still in flux.

Q&A


igalia