Greenfield

An update

Erik De Rijcke
erik.de.rijcke@udev.be
@friedchicken@mastodon.social

greenfield.app
wayouttheresoftware.blogspot.com
github.com/udevbe/greenfield
What is Greenfield?

- A Wayland compositor running entirely in your browser.
- Controls remote (X)Wayland applications from different servers.
  - Fast enough for games!
- Runs Wayland applications compiled to WebAssembly directly in your browser.
  - Also JavaScript applications.
Why is Greenfield?

- Wouldn't it be cool if we had...
  - A true ubiquitous personal computer.
  - A single interface for all your applications and data.
  - No attachment to a single piece of physical hardware.
  - No attachment to a single provider/SaaS.
How is Greenfield?

- Browser: TypeScript, C (WebAssembly) and WebGL.
- Server: TypeScript (I’m so sorry) and C.
  - I promise to rewrite it in Rust okay?
  - Also has a specialized implementation for k8s.
- Blood, sweat & tears.
A complicated picture about remote rendering
The roundtrip latency problem

- Need for a truly async protocol.
- Predictive frame callbacks.
- Fast sync.
- Good enough for gaming.
“Classic” Wayland Display Sync

Requests

Sync request

\[
\begin{array}{c}
R0 \\
R1 \\
R2 \\
R3 \\
R4 \\
R5 \\
\end{array}
\]

Events

Sync request always round-trips to browser and can only reply after events emitted by previous requests.

\[
\begin{array}{c}
E -1 \\
E 0 \\
E 4 \\
\end{array}
\]

Greenfield’s Fast Display Sync

Requests

Sync request

\[
\begin{array}{c}
R0 \\
R1 \\
R2 \\
R3 \\
R4 \\
R5 \\
\end{array}
\]

Events

Sync request is intercepted by Greenfield’s compositor proxy as it knows which previous requests will never send reply events.

\[
\begin{array}{c}
E -1 \\
E 0 \\
E 4 \\
\end{array}
\]

Sync done event is sent immediately, avoiding network round-trip
WebAssembly Applications in a bullet list

● Greenfield SDK.
  ○ Based on Emscripten.
  ○ Somewhat POSIX compatible.
    ■ Need a Linux WASM port.
● SHM buffers.
● WebGL through mesa (WIP).
WebAssembly Applications in a green diagram
Future

- Sound.
- Implement more Wayland protocols.
- Unified remote/browser/local filesystem.
- WASM32 port of Mesa to WebGL.
  - WebGL Wayland protocol already exists.
- Linux kernel port to WASM32.
  - ASM.js POC already exists.
  - This sh*t is hard yo!
  - But think of the possibilities!
A yellow diagram of a possible Linux WASM32 port
Another yellow diagram but of a possible single user remote file system
Hammer Demo Time!
You have questions?

I have answers!

Erik De Rijcke
erik.de.rijcke@udev.be
@friedchicken@mastodon.social

greenfield.app
wayouttheresoftware.blogspot.com
github.com/udevbe/greenfield