wlroots frame scheduling

A tale of disaster and chaos
Put the funny here

- I spent summer Doing Useful Work On Software 👍👍
- I am going to talk to you about it
- If you are a wlroots user then pay very close attention
The Endless Cycle

- Outputs (typically) refresh at fixed intervals
- The real world does not!
- Rendering takes variable time
- Client rendering takes variable time but in a much more sinister way (compositor-specific)

```
client render
0 1 2 3

rendering
0 1 2 3

user input
• • • • • • • •
```
wlroots in days gone by

- Most wlroots users have this timeline
- It’s kind of encouraged by wlroots’ interface and examples
- Frame callbacks are sent when rendering starts (oh no!)

```
client render
  0  1  2  3
rendering
  0  1  2  3
user input
  • • • • • • • •
```
Ideally

- Rendering finishes just in time to present – no wasted time
- Can collect both user inputs and client frames
- Less latency!

```
client render: 0 1 2 3
rendering:     0 1 2 3
user input:    • • • • • • • • •
```
Young and Naive

- First patch tried to add pre-render delay to the existing frame scheduling
- I thought I knew it all
- Learning by failure
- The GSoC “get your bearings” period is during my exams
- Thanks Kenny love you
Winning the Lottery

• We want to finish rendering as close to the deadline as possible
• We can’t know how long rendering will take… but we can learn from previous frames
Timer? I hardly know her!

- Next on the agenda is render timers
- If we want to learn from previous frames, we better have measured them
- `wlr_render_timer`, yay
Winning the Lottery

- We want to finish rendering as close to the deadline as possible
- We can’t know how long rendering will take... but we can learn from previous frames
- If only we had a tool that could show us, the developers, frame time behaviour
- That sure would be neat
The War on Computers

- Linux has a tracing system that can show us GPU buffer events and such
- Through `/sys/kernel/tracing/user_events_data` we can inject events into it. Seems appropriate!
- Enter `libuserevents`!
- `/sys/kernel/tracing/` hates me
- I ran out of time
- No more frame time tool
- `libuserevents` still exists though
Unforeseen Consequences

- Turns out writing compositors is hard
- `wlr_output_schedule_frame` makes a frame event happen “at some point”
- This is a lot more complicated than it sounds
Success, finally?

- Say goodbye to `wlr_output_schedule_frame`
- Enter the new frame scheduler interface
- The interface abstracts over presentation-feedback-based scheduling, `wl_surface.frame`-based scheduling, etc.
- Along with the render timer interface I think this is the only part of the project that didn’t fail 😎
- MR !4307 genuinely sitting on the finish line right now
- One day we will have an implementation that supports firing frame a certain interval after present! (one day...)
But Not Today

- In the meantime, users get to handle the breakage of new frame scheduling for no benefit :)
- Enjoy your new and shiny render timers
- Thanks to Simon for putting up with me and my fullbright forehead
- Thanks to you for coming!
- Pray you don’t see me again next summer