Rendering Plans for 2022

A Biased View

Stefan Zager, Google
szager@chromium.org

On the Cusp of Rendering NG

Rendering NG Projects

Composite After Paint (CAP), 2014 - 2021

Next Generation Layout (LayoutNG), 2016 - 2022

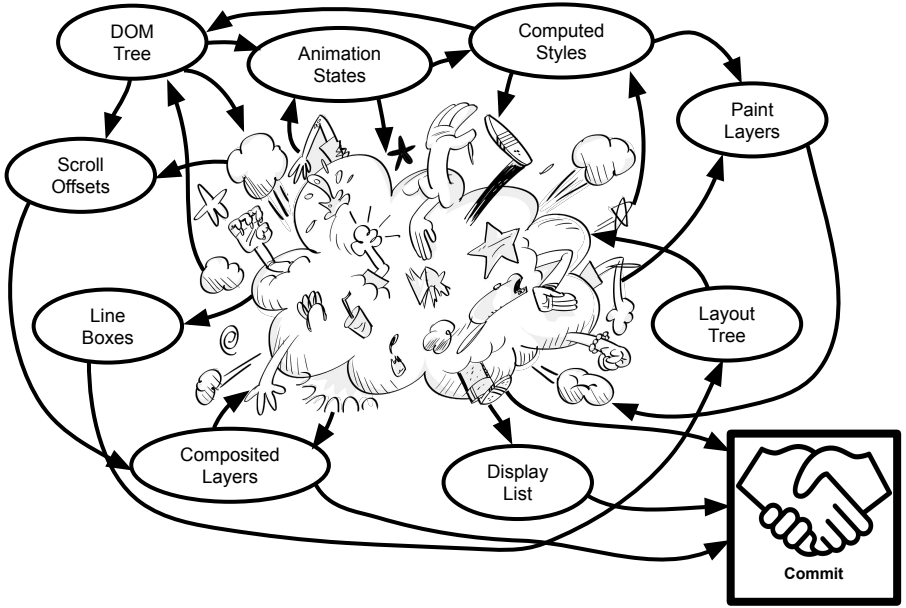
Squad, 2018-2019

Rendering NG Guiding Principles

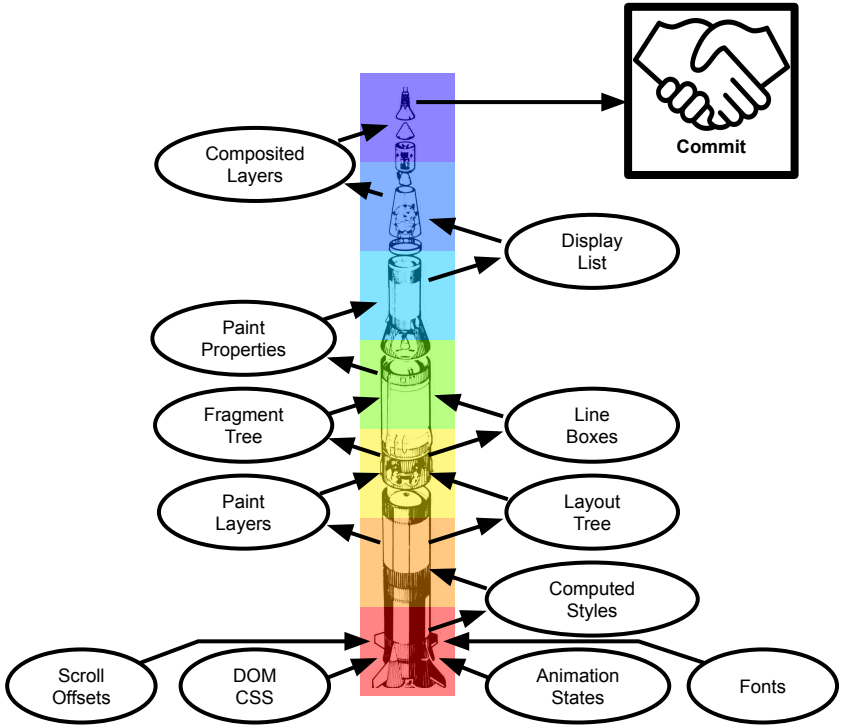
All rendering stages should be **functional**: defined inputs, defined outputs, deterministic behavior.

All rendering stages should be **WORM**-y (**W**rite **O**nce, **R**ead **M**any): the output of any rendering stage must be immutable once the stage is finished.

Before



After



2022 Goal #1: Finish Rendering NG

The last big piece of LayoutNG is support for fragmentation (multi-column, pagination, etc.), expected to ship in H1.

Massive code cleanup and low-hanging performance fruit.

Feature Work

Form Controls

Anchor Positioning
(Position one thing relative to another)

(No prototype yet - still working on API shape)

<selectmenu>
(the new element itself)

(Prototype in Chromium)

<popup>
(Always-on-top, one at a time, and light-dismiss)

(Prototype in Chromium)

CSS Container Queries

```
<style>
.top {
  container-type: inline-size;
  container-name: top;
  margin: 10px;
  border 1px solid blue;
}
.flexbox {
  display: flex;
}
@container top (max-width:100px) {
  .flexbox {
    flex-direction: column;
  }
}
</style>
<div class=”top” style=”width:100px”><div class=”flexbox”>
  <img><span>Lorem ipsum dolor sit amet</span>
</div></div>
<div class=”top” style=”width:200px”><div class=”flexbox”>
  <img><span>Lorem ipsum dolor sit amet</span>
</div></div>
```

Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet

Fluid and Flexible UI

Broad initiative to deliver high quality modern UI components for the web.

For more information: unakravets@ and nsull@

Rendering Performance

Key Insights

Better rendering performance == lower battery consumption

Rendering pipeline accounts for ~50% (!) of all renderer process CPU utilization on Android.

Rendering “smoothness” is key to users’ perception of quality

Ongoing work to define and measure smoothness:
https://web.dev/smoothness/

Near-Term Performance Projects

Incremental compositing improvements (e.g. compositor-driven blinking caret, compositor-driven animated GIFs)

Threaded text shaping

Non-Blocking Commit (crbug.com/1255972)

Off Main Thread Compositing (crbug.com/1228581)

Long-term Performance Projects

Off Main Thread Paint

Off Main Thread Pre-Paint

Hit Testing the Fragment Tree (off main thread?)