

Are We LBSE Yet?

Closing the Gaps on WebKit's New SVG Engine

WebKit Contributors Meeting 2026 · October 20–21, 2026

Nikolas Zimmermann

nzimmermann@igalia.com

About me & Igalia



Nikolas Zimmermann
Aachen, Germany

- 2001** Co-created KSVG2 for khtml
- 2005** WebKit contributor
- 2007** WebKit reviewer
- 2019** Joined Igalia



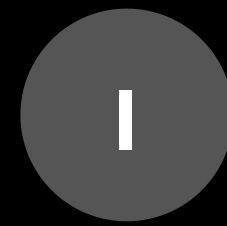
igalia

Since 2001 · A Coruña, Spain

- › Worker-owned open source consultancy
- › Maintains WebKitGTK & WPE WebKit
- › Works on WebKit, Chromium, Gecko & Servo
- › Funded the 2026 LBSE work



Outline



LBSE in a nutshell

The idea and the road so far



A new composition model

Conditional layers & paint order segments



Gap 1: performance

MotionMark Suits, April vs. October 2026



Gap 2: features & fixes

What we fixed this year



Are we LBSE yet?

Where we stand, what's needed for the default



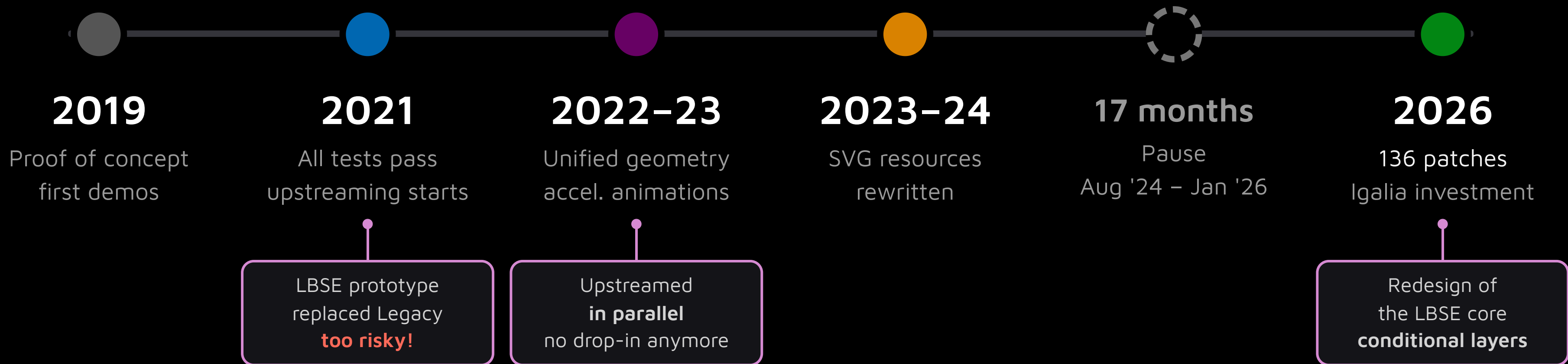
I What is LBSE?

Layer-Based SVG Engine — the new SVG engine in WebKit

- SVG can take part in the **RenderLayer tree**
 - hardware-accelerated **compositing** and **accelerated** transform animations
 - **3D transforms**, perspective, **z-index** for SVG content
- **Shares the HTML/CSS machinery** only where an element needs it
 - **NEW** plain shapes & untransformed groups paint **without a layer**, in DOM order
 - **NEW** shared building blocks: **ClipPathPaintScope**, **TransformPaintScope**
 - SVG filters run through **RenderLayerFilters**, like CSS filters
- Developed **in parallel** to the legacy SVG engine
 - a **runtime-enabled feature**: switch engines on the same binary



A long road



~290 patches since 2021 · tracked in <https://github.com/nikolaszimmermann/WebKitIgalia/issues/1>



A new composition model

- **Problem:** every SVG renderer had its own RenderLayer — simple, but far too wasteful (ruins MotionMark performance)
- **Solution:**
 - 1. Conditional layer creation:**
a layer only where it is needed
 - 2. Paint order segments:**
correct paint order with mixed composited / plain children



A layer per renderer

- Early LBSE: `RenderSVGModelObject::requiresLayer()` → `true`
 - the layer tree orders children, applies transforms, clips, composites — **for free**
 - DOM paint order guaranteed, `z-index` still available
- Never meant to ship as-is: on Suits, **thousands of layers**
 - each one takes part in per-frame bookkeeping: position updates, z-order lists, transform updates, repaint walks
- Goal: give a renderer a `RenderLayer` **only when it actually needs one**

Blog: **Reducing layer overhead in LBSE** (July 14, 2026)

<https://blogs.igalia.com/nzimmermann/posts/2026-07-14-lbse-conditional-layers/>



Conditional layer creation

New homes for layer state

- SVG-only layer fields → lazy `SVGData`
- transform cached on the renderer

DOM-order paint & hit testing

- DOM-order list of renderable children
- paint & hit-test paths without a layer

Still gets a layer

- opacity, filters, blend modes, transformed containers
- compositing, 3D, perspective
- clip-path & mask (until August)

No layer anymore

- plain shapes, untransformed groups
 - leaves with a 2D transform
- **the bulk of every drawing**

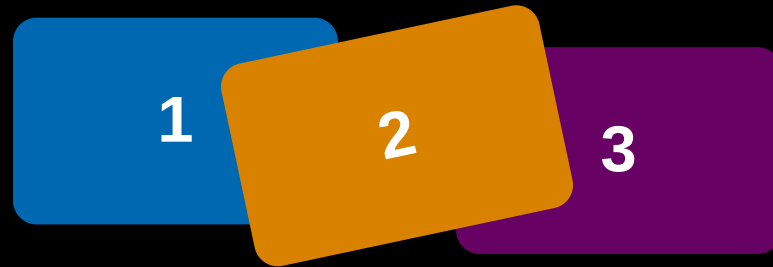
A chain of dormant preparation patches, then the switch: `315610@main` (June 22, 2026)



SVG paint order

HTML

- 3 `<div>`s, middle one transformed
- a transform creates a stacking context



→ the middle one paints last, on top

SVG

- strict **document order**
- with or without a layer



→ the middle one stays in between

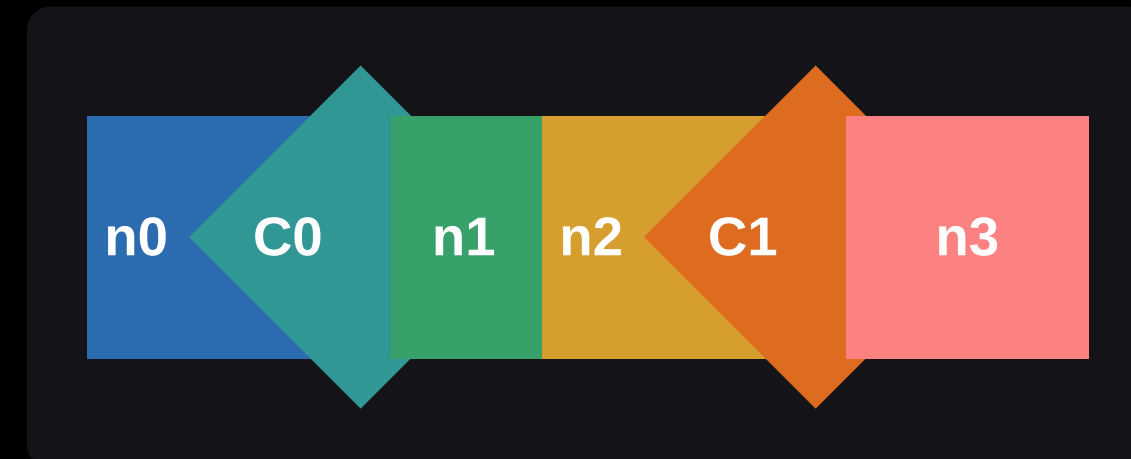
- Early LBSE: one layer per renderer, paint order maintained by the layer tree
- With **mixed** layered / plain children, LBSE needs to maintain document order
- Hardest case: **composited** children next to plain ones



Compositing vs. paint order

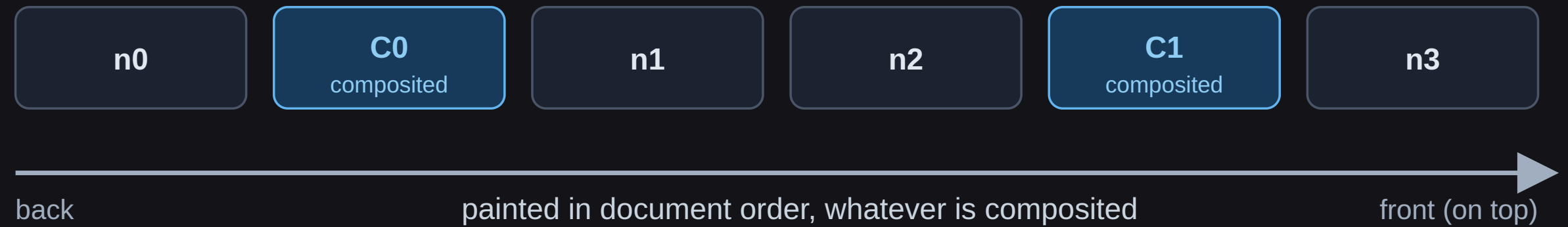
Consider the example of a simple SVG document, that has two compositor-animated shapes interleaved with regular shapes:

```
<g>
  <rect id="n0"/>
  <rect id="C0" class="spin"/> <!-- composited -->
  <rect id="n1"/>
  <rect id="n2"/>
  <rect id="C1" class="spin"/> <!-- composited -->
  <rect id="n3"/>
</g>
```

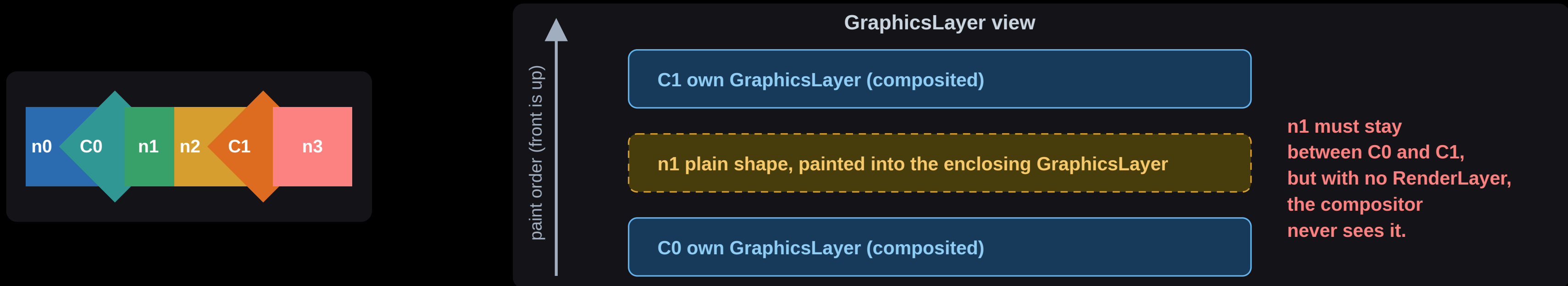


Desired rendering: document order, with **C0** and **C1** composited (accelerated animation)

DOM view



Composited vs. plain siblings



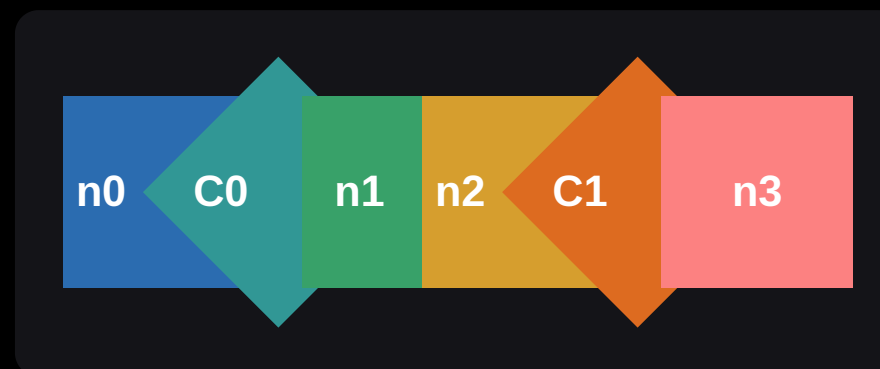
- `n0` ... `n3` paint into the backing store of the **nearest enclosing GraphicsLayer**
- The compositor can put that layer in front of `C0` and `C1`, or behind both — **never in between**

Blog: **Implementing compositing in LBSE** (July 22, 2026)

<https://blogs.igalia.com/nzimmermann/posts/2026-07-22-lbse-paint-order-segments/>



First attempt: promotion



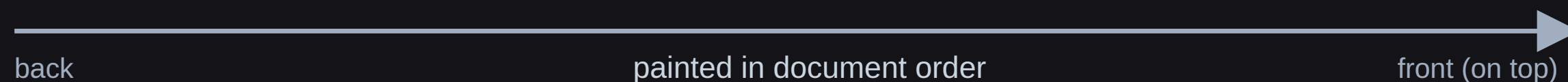
1. Layer creation, at style time. Every sibling after a boundary is given a RenderLayer.



2. Compositing pass. Every sibling after the first composited boundary is promoted to composited.



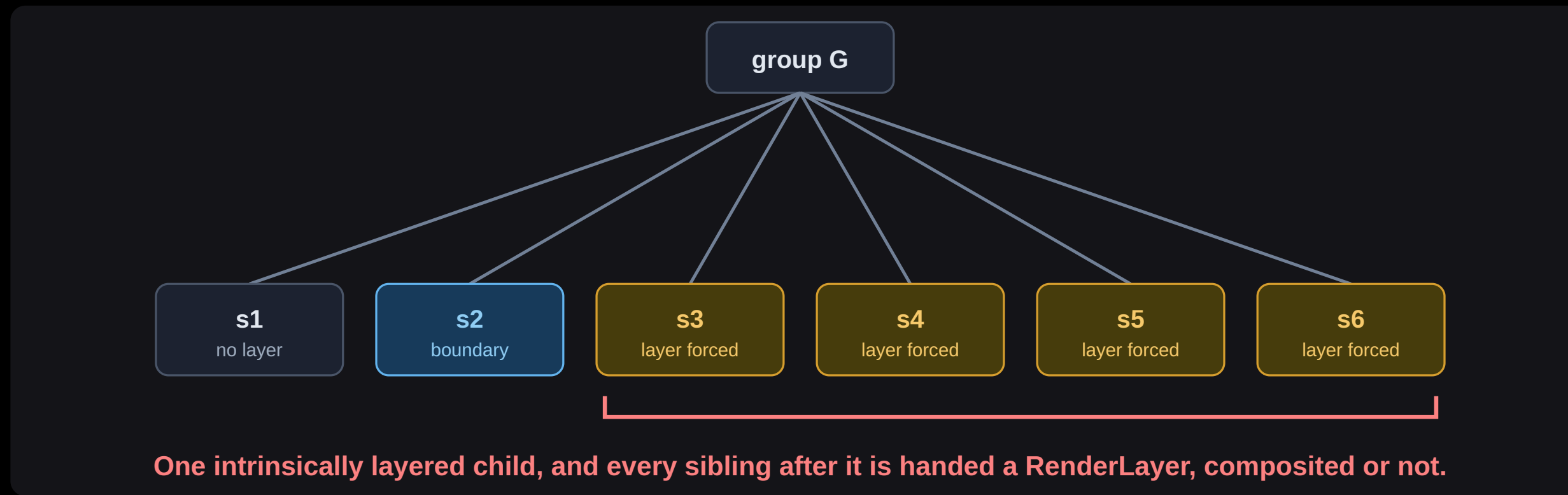
Only n0, which comes before C0, escapes.
Everything after it ends up with a RenderLayer, then a GraphicsLayer.



It worked, all tests passed — **but** step 1 runs at style time, before any compositing decision exists



First attempt: the catch

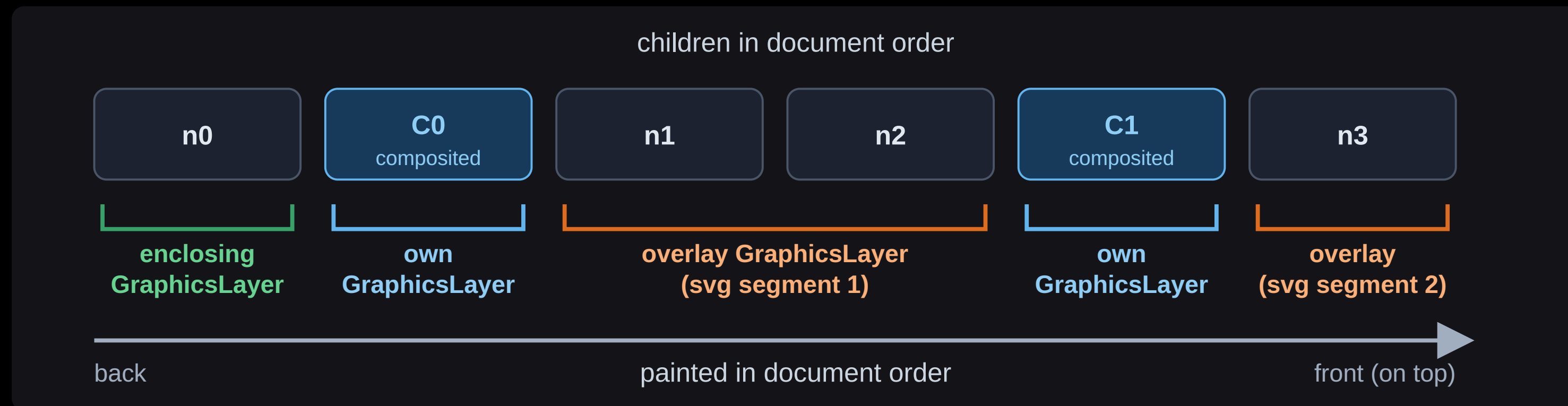
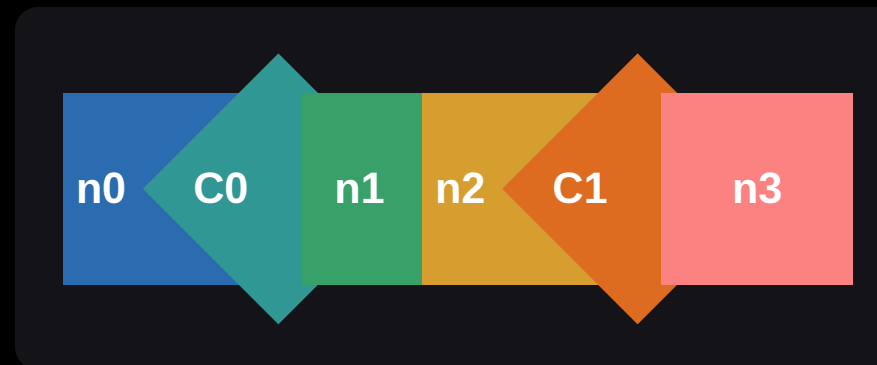


Exactly the layers we had just spent half a year removing → **thrown away!**

Need to look for a **better solution**, without performance cliffs that depend on which element in the DOM needs compositing.



Solution: paint order segments



- Split the children into **segments** at every composited sibling, one **overlay GraphicsLayer** per segment
- Computed **after** compositing decisions: no compositing → no segments, no overlays, no extra RenderLayers
- Landed as [315674@main](#) (June 23, 2026)



III Gap 1: performance

- › **Groundwork:** conditional layers forced a new composition logic — all preparation for the performance work
- › **April 2026:** LBSE **26.8%** behind Legacy on MotionMark Suits
- › **October 2026:** how close did we get?



Why Suits?



1.3.1 · Suits

- A flat `<svg>`: **clipped** `<rect>`s and `<path>`s, each with a **gradient**
- Every frame, the `transform` attribute of **every shape** changes
- Score: how many shapes the engine keeps moving at 60 FPS
- **Avoids** the compositing fast path → LBSE has the **least** to offer
- April 2026: the **largest gap** to Legacy of all benchmarks **we track**

Blog: **Making LBSE performant** (October 21, 2026)

<https://blogs.igalia.com/nzimmermann/posts/2026-10-21-lbse-performance/>



Layers in Suits

Until June 2026

```
svg layer
  rect clip-path layer
  path layer
  rect clip-path layer
  path layer
  ...
```

every renderer gets a RenderLayer

June 2026

```
svg layer
  rect clip-path layer
  path
  rect clip-path layer
  path
  ...
```

conditional layer creation

August 2026

```
svg layer
  rect clip-path
  path
  rect clip-path
  path
  ...
```

clip-path without a layer



A fair comparison

Same binary

- one release build per revision
 - the engine is a runtime setting
- **both engines, same build**

Interleaved runs

- one discarded warmup per engine
 - then Legacy, LBSE, Legacy, LBSE, ...
- **3 scored runs per engine**

Clean runs

- all WebKit processes killed first
- minimized background activity

Converged ramps only

- interval $\geq \sim \pm 10\%$ → run dropped
- **each point = median of 3 runs**

REVISIONS **62** · Apr 17 → Oct 5, 2026

BROWSER MiniBrowser · fullscreen

BENCHMARK MotionMark 1.3.1 **Suits** · ramp, 60 s

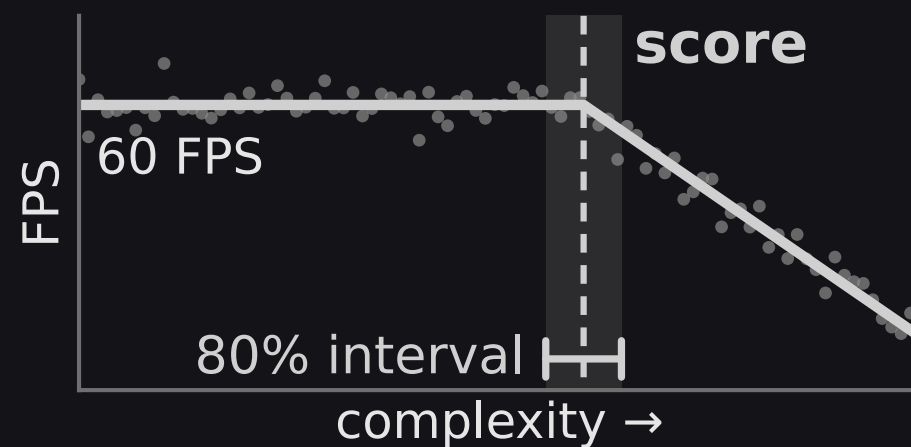
MACHINE Apple M4 Max · macOS 26.5



What does one MotionMark run tell us?

1 · ONE RUN

Score: highest complexity still at 60 FPS. **80% interval** (as MotionMark reports it): how precisely the ramp pins it down.



I_{one} : $\pm 1.1\%$ $\pm 2.4\%$
LBSE LEGACY

Typical interval, from >150 runs per engine.
 $\geq \pm 10\%$: ramp did not converge, run dropped.

2 · N RUNS

Noise study: 5 “calm” runs × M revisions.
Gaussian-like scatter:

$$\sigma_{\text{run}} \approx 1.5\% \text{ (LBSE)},$$
$$\sigma_{\text{run}} \approx 1.4\% \text{ (Legacy)}.$$

Every Gaussian has 80% within $\pm 1.28\sigma$ (and 95% within $\pm 2\sigma$):

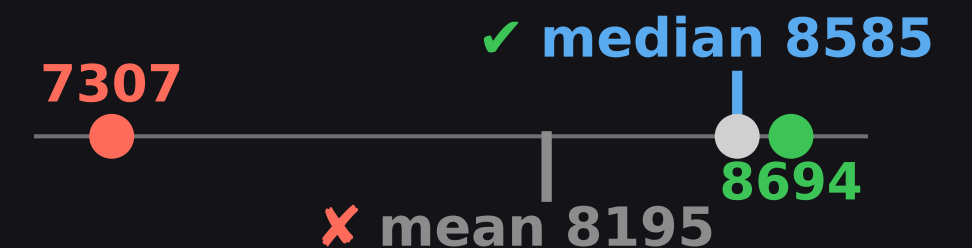
$$\rightarrow I_{\text{many}} = \pm 1.28 \cdot \sigma_{\text{run}} \approx \pm 1.28 \cdot 1.5\%$$
$$\approx \pm 1.9\% \text{ (Legacy } \pm 1.8\%)$$



One run can't see the noise between runs.
But: the **mean** of N runs shrinks I_{many} by $\sqrt{N}$.

3 · USING THE MEDIAN

Problem: converged “non-calm” runs
may score too low: 8585 · 8694 · 7307



Solution: each point = **median** of 3 runs
(in all following LBSE vs Legacy plots).

But which uncertainty for each point?

Measure scatter σ_{point} in ~20 “quiet” revisions,
“non-calm” runs included:

	LBSE	Legacy
1 run	1.55%	3.89%
2 runs	1.34%	2.83%
mean of 3	1.26%	2.34%
median of 3	1.19%	1.52%



Is a change between revisions real?

1 · SUMMARY: ONE POINT

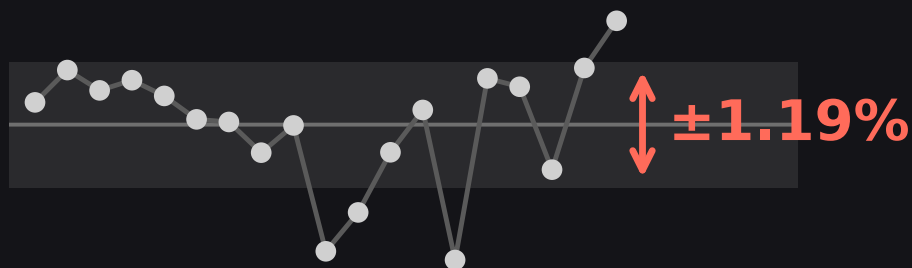
1.19% **1.52%**

LBSE

LEGACY

σ_{point} : how much a point (**median of 3**) scatters across “**quiet**”, but not necessarily “**calm**” revisions.

E.g. LBSE, ~20 “**quiet**” revisions:



“**quiet**”: no perf-relevant change landed.

“**non-calm**”: the 3 runs may contain outliers.

2 · ONE STEP

A **step** is the change of one engine from one revision to the next. Both points fluctuate **independently**:

$$\begin{aligned}\sigma_{\text{step}} &= \sqrt{\sigma_{\text{point}}^2 + \sigma_{\text{point}}^2} \\ &= \sqrt{2} \cdot \sigma_{\text{point}}\end{aligned}$$

$$\text{LBSE: } \sqrt{2} \cdot 1.19\% \approx 1.68\%$$

$$\text{Legacy: } \sqrt{2} \cdot 1.52\% \approx 2.15\%$$

Note: neighbouring points are slightly correlated.
Ignored for brevity: the true range is slightly smaller.

3 · 95% RANGE

95% of noise stays within $\pm 2\sigma$.

Why 95%? A step counts as real only if noise alone would rarely produce it.

A step's range is $\pm 2 \cdot \sigma_{\text{step}}$:

$$\text{LBSE: } \pm 2 \cdot 1.68\% \approx \pm 3.36\%$$

$$\text{Legacy: } \pm 2 \cdot 2.15\% \approx \pm 4.30\%$$

±3.36% **±4.30%**

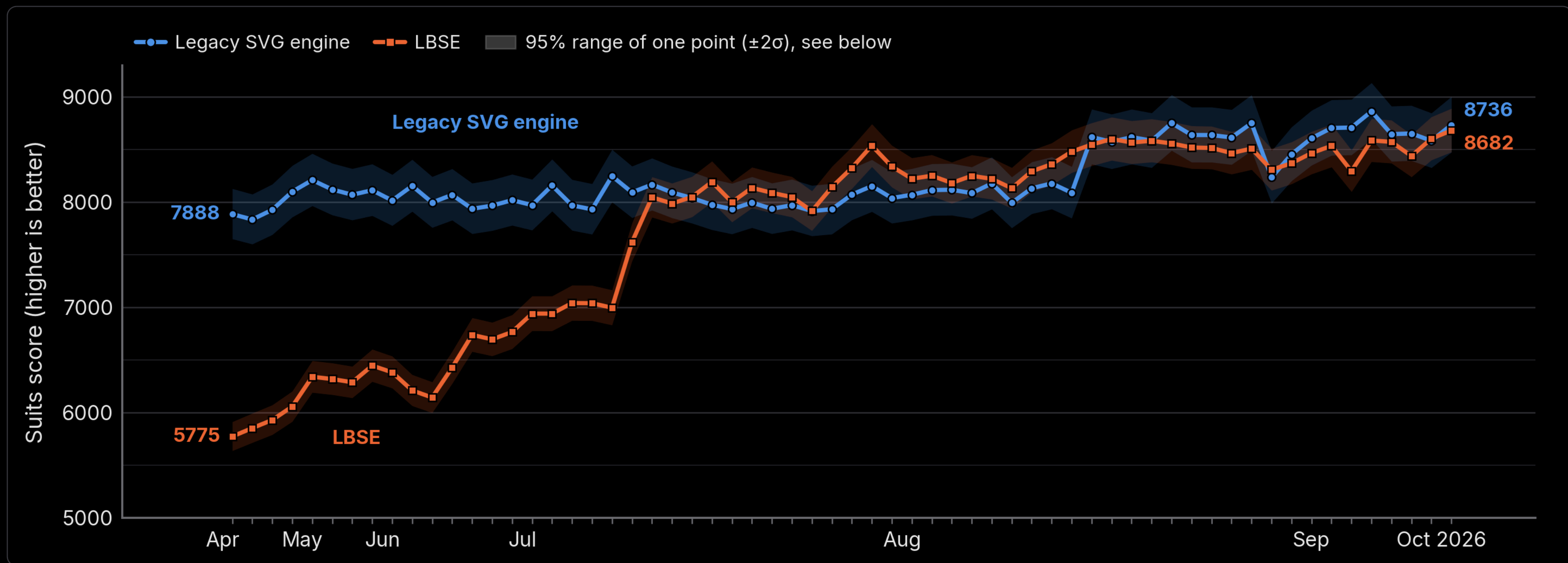
LBSE

LEGACY

A step between **two revisions**, looked at **in isolation**: if it is smaller than this, it can't be told apart from noise. Averaging can improve the sensitivity.



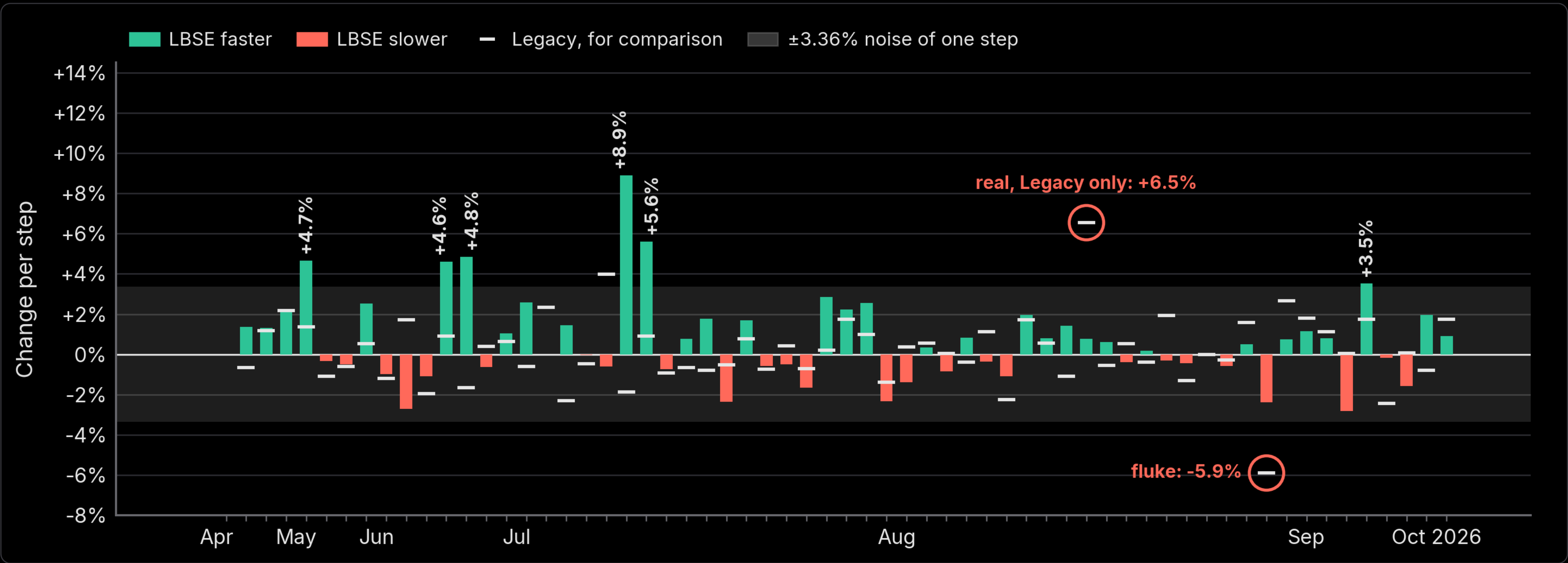
April → October 2026



Shaded: 95% noise range of one point, $\pm 2\sigma_{\text{point}}$: LBSE $\pm 2 \cdot 1.19\% \approx \pm 2.38\%$, Legacy $\pm 2 \cdot 1.52\% \approx \pm 3.04\%$



Changes per step



Each column: change in the LBSE score from the previous measured revision · labelled: steps outside the $\pm 3.36\%$ noise



The patches that stand out

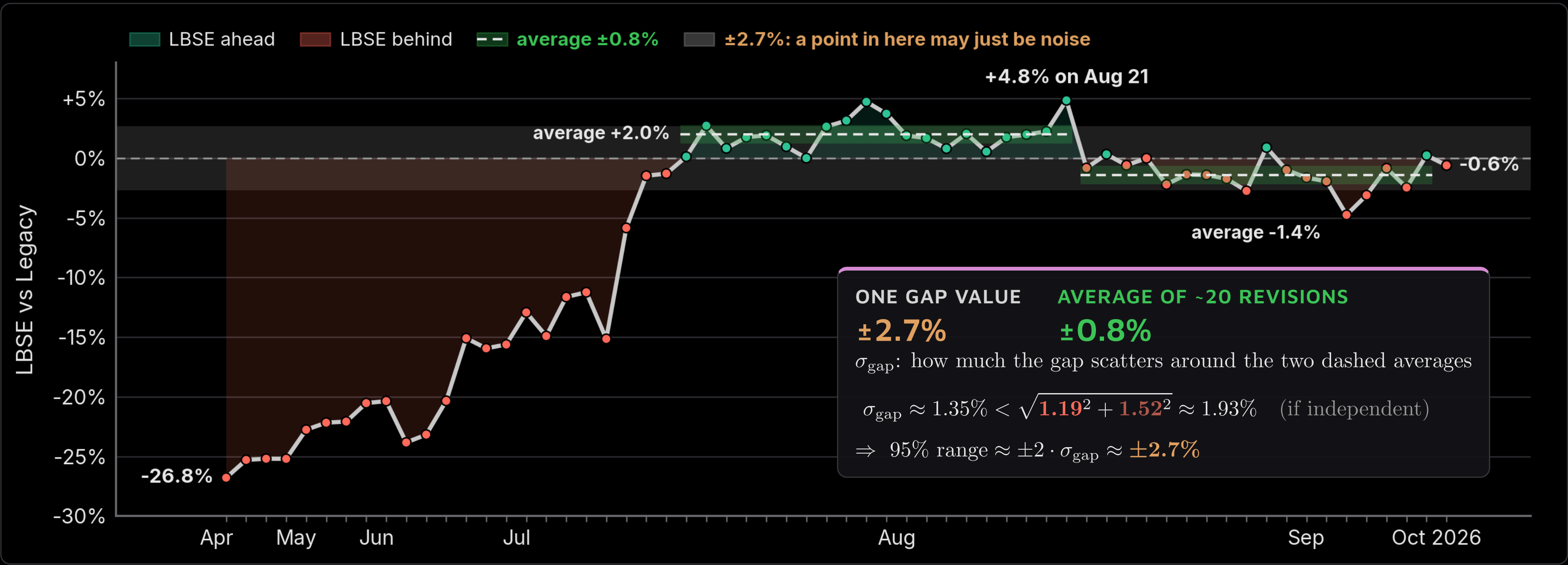
When	Revision	Patch	Step
Apr 21	311726@main	Suppress LayoutRepainter in hidden and anonymous containers	+1.4%
Apr 27	312090@main	Cache the local transform of non-layered elements	+1.3%
May 19	313491@main	Skip the redundant fill clip for rects clipped inside their bounds (from Legacy)	+4.7%
Jun 3	314458@main	Defer per-element transform attribute work (one flush per frame)	+2.5%
Jul 2	316428@main	Cache the resolved fill / stroke paint server	+4.6%
Jul 6	316543@main	Cache the concatenated transform attribute matrix	+4.8%
Jul 6	316545@main	Cache the viewport size for the transform reference box	+1.0%
Jul 10	316880@main	Coalesce the transform flush into one repaint per container	+2.6%
Jul 13	317077@main	Cache the transform origin	+1.4%
Aug 1	318375@main	No RenderLayer for SVG clip-path	+8.9%
Aug 2	318410@main	Skip the outline paint pass for outline-free renderers	+5.6%
Aug 3	318440@main	Skip the per-frame visual-overflow recompute when culling	+1.8%
Aug 10	318882@main	Cache the paint server on the renderer	+2.9%
Aug 18	319390@main	No RenderLayer for SVG mask (Suits has no masks)	+0.8%
Oct 5	322676@main	Avoid redundant save/restore pairs (8.5 → 6.5 IPC messages per shape)	+0.9%

Grey steps are within the $\pm 3.36\%$ noise of a single step.

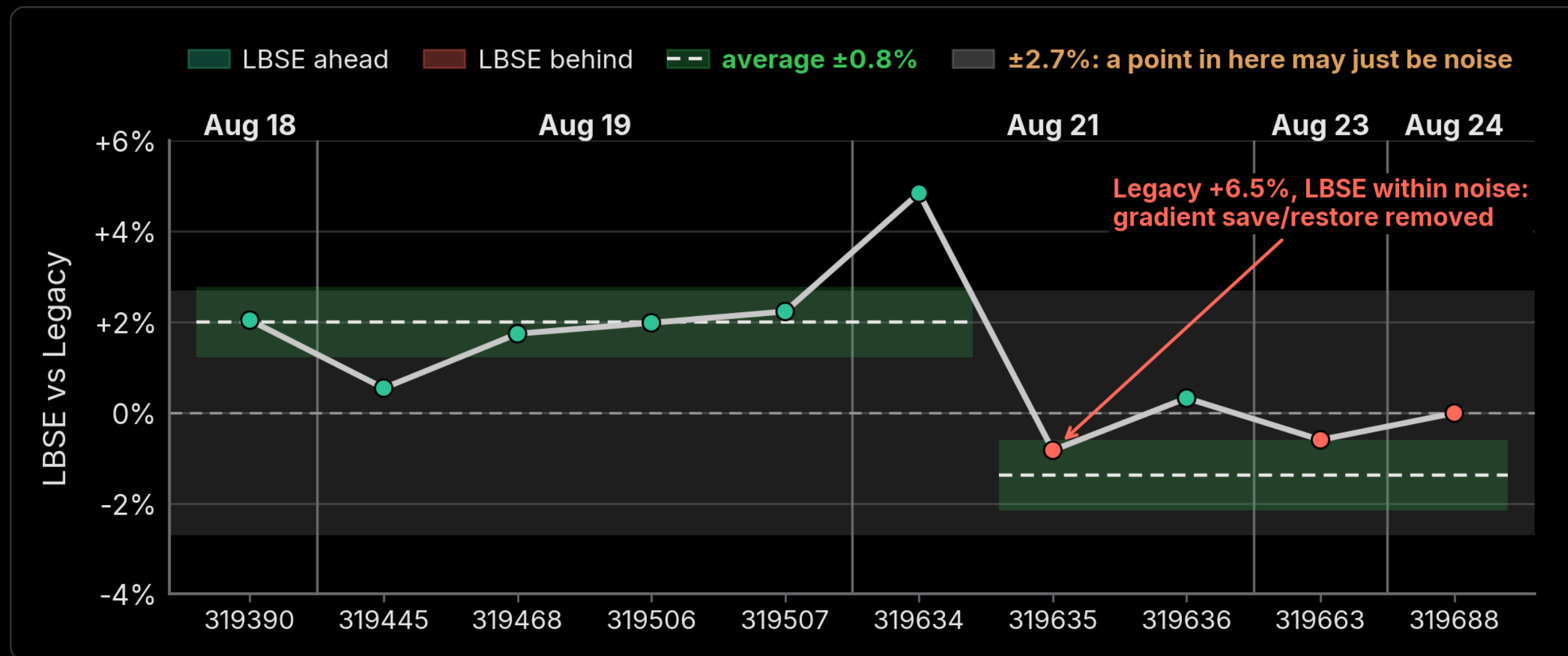
Not listed: +3.5% on Sep 30. No SVG change landed there. The revision before it (Sep 29, 72 commits earlier) just measured low.



The gap over time



Why the August lead went away

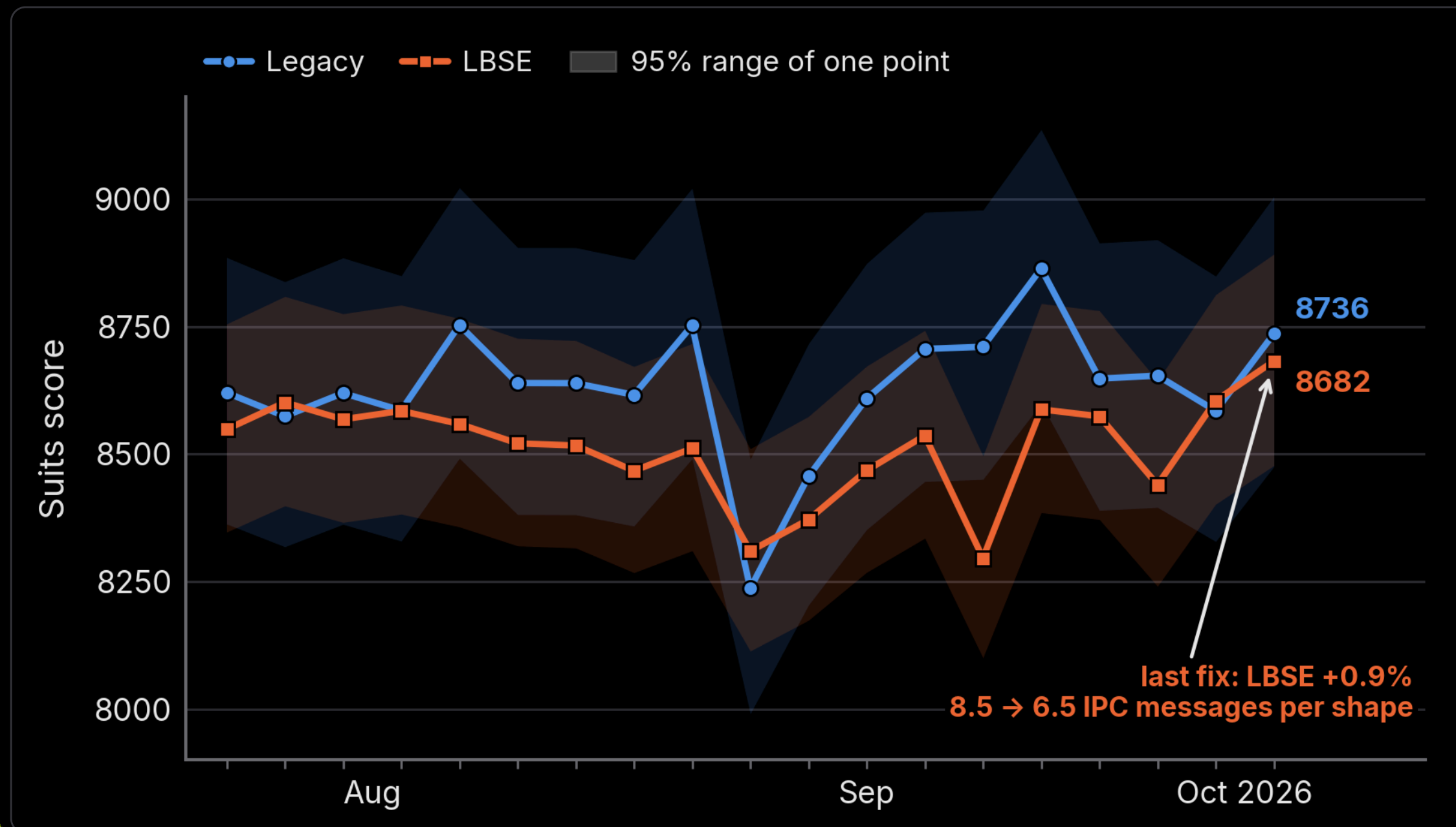


Legacy only: +6.5%, one commit

Remove redundant save/restore around SVG gradient fill/stroke (319635@main). Each save and restore is an IPC message from the WebContent to the GPU process: two per gradient-filled shape and frame, which LBSE never sent. LBSE stayed within the noise.



October 2026: how close did we get?



+50%

LBSE since April

LBSE 5775 → 8682 points +50.3%

Legacy 7888 → 8736 points +10.8%

-26.8%

gap in April

LBSE behind Legacy

-1.4% ±0.8%

gap Aug 21 – Oct 3

average of 18 revisions, before the last fix

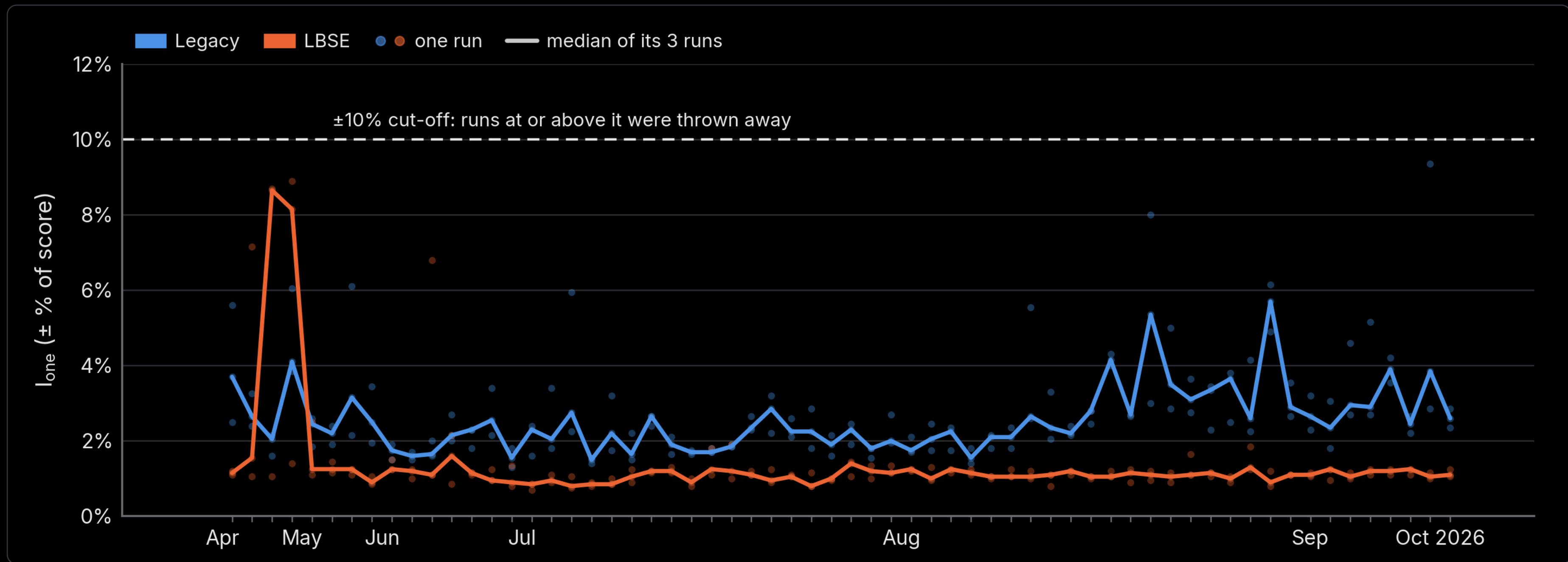
-0.6%

gap at the latest revision (Oct 5)

last fix: LBSE +0.9%, within the noise of one step



A nice side effect: steadier frames



Median MotionMark confidence interval per run (l_{one}): $\pm 2.4\%$ Legacy vs. $\pm 1.1\%$ LBSE · 69 of 71 non-converged ramps were Legacy runs

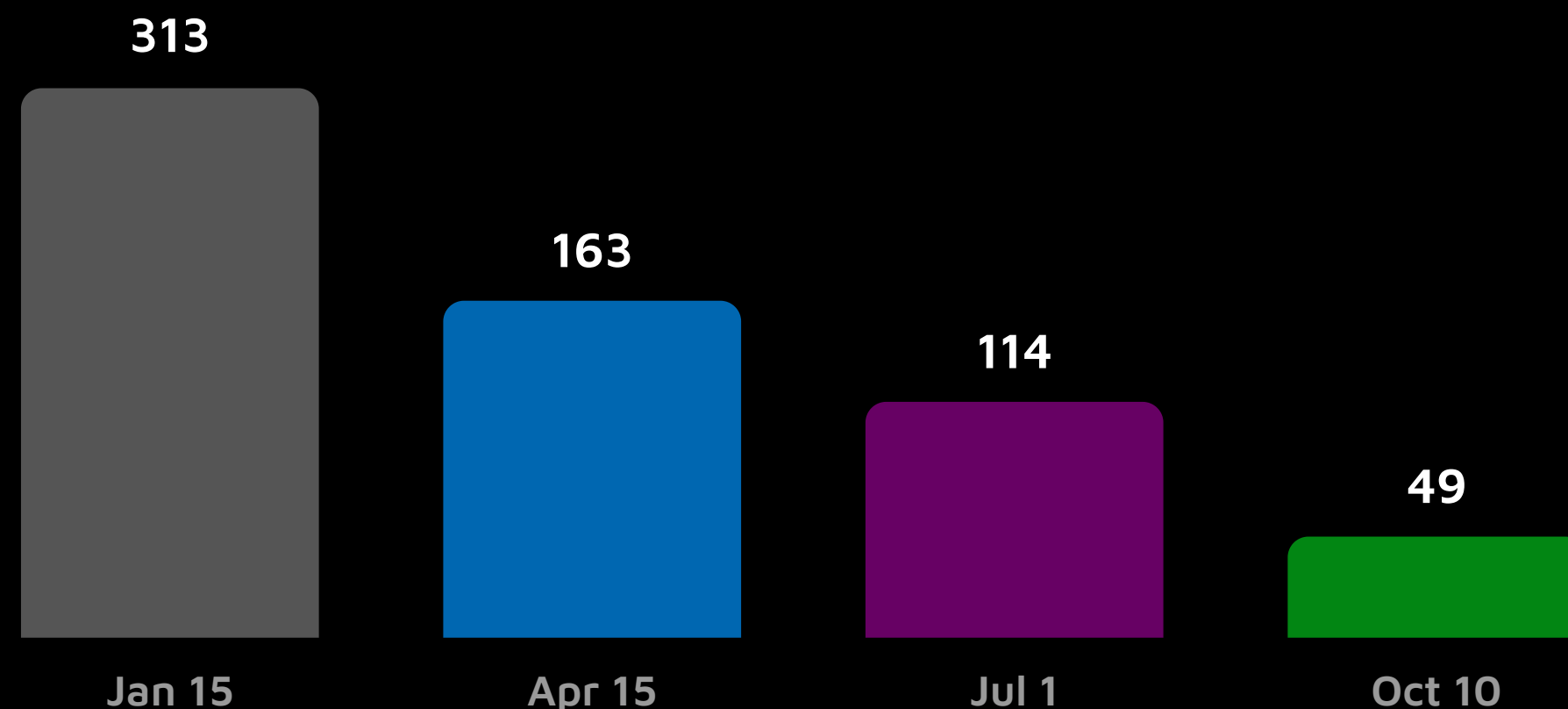


IV Gap 2: features & fixes

- › **January 2026:** after 17 months without any testing:
 - text did not render at all (fixed in `305312@main`)
 - most test baselines were outdated
 - Legacy fixes that had never reached LBSE
- › **October 2026:** expected test failures under LBSE **313** → **49**,
crashes **3** → **0**
- › **Fixes:** dozens of them: debug assertions (all but 1),
crashes, text, filters, clipping, masking,
outermost `<svg>`, ...



Expected failures under LBSE



Failure, ImageOnlyFailure and Crash entries in `platform/mac-tahoe-wk2-lbse-text/TestExpectations`
(`css3/masking`, `imported/w3c/web-platform-tests/svg`, `svg`, pixel tests at tolerance 0)



Fixed in 2026

Crashes & hangs	getBBox on <code><use></code> · NaN paths with line caps · marker child changes · <code>xlink:href</code> cycles · namespaced resources
Text	not rendering · opacity · blurry when zoomed · glyph hit testing · <code>pointer-events</code> · <code>bounding-box</code> · <code><textPath></code>
Filters	invalid filters · <code>feOffset</code> clipping · scaling · on the outermost <code><svg></code> · <code><feImage></code> via the layer tree · filter region
Clipping & masking	mask issues · cut-off masked content · transformed <code>objectBoundingBox</code> <code><clipPath></code> · external and <code>data:</code> URLs
Outermost & nested <code><svg></code>	view fragments · empty and animated <code>viewBox</code> · <code>transform</code> on the root · scroll size · nested <code><svg></code> clipping
Repaint & hit testing	stale repaint rects after transforms · HTML clients of SVG resources · hit testing in transformed groups
Gradients, patterns, markers	non-invertible <code>gradientTransform</code> · text paint-order with gradients · dynamic markers · additive <code>animateMotion</code>
Strokes	<code>non-scaling-stroke</code> : wrong coordinate system, stale width after transforms
Conditional layers fallout	event regions of non-layered children · paint order around composited siblings · 2D transform flags
Still in review	<code>mix-blend-mode</code> on the root · hidden <code><text></code> / <code><foreignObject></code> with visible children · repaints in groups

136 LBSE patches in 2026, about 100 of them fixes like these · full list: github.com/nikolaszimmermann/WebKitGalia/issues/1



Are we LBSE yet?

- › An honest look at **where we stand**
- › What we think is **needed to make LBSE the default**



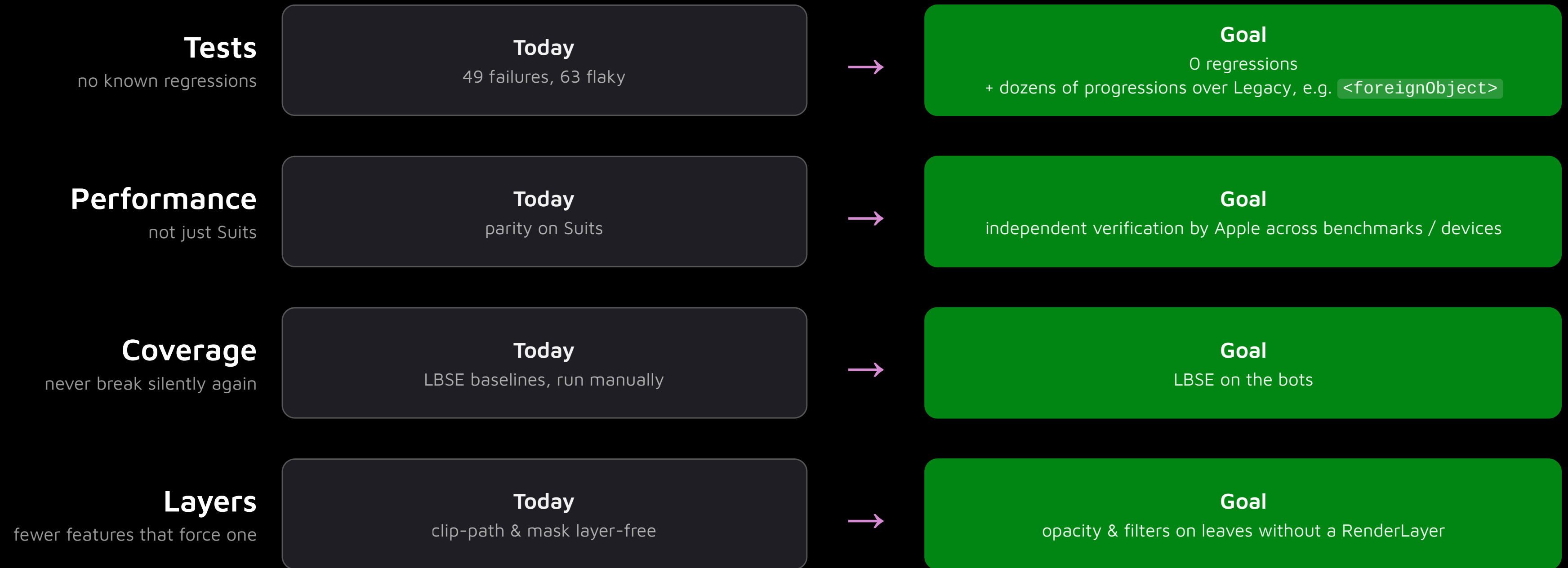
Are we LBSE yet? Almost!

●	Composition	Layer-free by default · to be investigated: opacity , filters and blend modes still force a RenderLayer, even on leaves
●	Performance	Suits: -0.6% latest, pending confirmation Apple A/B test, mid-Aug: Speedometer 3 & MotionMark 1.3.1 perf-neutral
●	Feature parity	49 expected failures (29 of them filters)
●	Default	Not yet · needs independent verification and the green light from Apple

Status, October 2026 — ● done ● close ● not yet



The road to default



LBSE in 2026

136

patches in 2026

~290 since 2021

1

layer left in Suits

the <svg> root, not one per shape

-26.8% → -0.6%

gap to Legacy on Suits

April → latest, pending confirmation

313 → 49

expected test failures

crashes 3 → 0

Thanks to **Rob Buis** for reviewing most of these patches, to **Simon, Said, Karl** and **Ahmad** for the reviews and the hard questions, and to **lgalia** for funding this round of work.



Thanks!

Questions & discussion



