CSS pointer-driven animations: how pointer timelines work in the W3C draft

26 November 2026 · 4 min read · blogs.iictp.org/css-pointer-driven-animations-pointer-timeline-explained

Short answer: CSS pointer-driven animations are a proposed web feature in which the position of the pointer, not elapsed time, controls how far an animation has progressed. The W3C CSS Working Group published the First Public Working Draft of Pointer-driven Animations Module Level 1 on 24 November 2026, defining pointer(), pointer-timeline properties and a PointerTimeline interface. It matters because effects that usually need JavaScript pointer listeners could become declarative CSS.

TL;DR The short version

- Pointer-driven animations are a proposed CSS feature in which the pointer's position, not time, controls an animation's progress.
- The W3C published the First Public Working Draft on 24 November 2026, defining pointer(), pointer-timeline properties and a PointerTimeline interface.
- It is an early draft, so prototype behind feature detection, respect reduced motion, and keep JavaScript fallbacks.

Read online, listen to it, or change the text size: https://blogs.iictp.org/css-pointer-driven-animations-pointer-timeline-explained

What are pointer-driven animations?

The draft defines pointer progress timelines: timelines linked to the pointer’s position over an element’s box along one axis. The start of the range is 0% progress and the end is 100%. It builds on Web Animations and CSS Animations and says it was derived from, and inspired by, scroll-driven animations, which use the same idea of a timeline measured in distance instead of seconds.

How pointer timelines work

- **Anonymous timeline:** animation-timeline: pointer(<source> <axis>).
- **Source:** self (the default, the element’s own box), nearest (described as the nearest element’s parent box) or root (the root element’s box).
- **Axis:** block, inline, x or y. The pointer() function defaults to inline.
- **Named timeline:** pointer-timeline-name and pointer-timeline-axis, with the pointer-timeline shorthand. The initial value of the axis property is block.
- **Script access:** a PointerTimeline object in the Web Animations API.

What is new compared with script-based effects

Today a tilt, spotlight or parallax effect typically listens for pointer events, computes a percentage and writes styles on every frame. In the draft, the browser holds that relationship: you declare which box defines the range and which keyframes run across it. The same model as scroll-driven animations means one mental model for both.

Ranges defined in the draft

- fill: the full range, from the start padding edge to the end padding edge.
- fit: the same size as fill, shifted so that 50% progress stays at the range centre set by animation-range-center.
- cover: the largest range around that centre which fully covers the element’s box.
- contain: the largest range around that centre which stays fully inside the box.

Limitations and open questions

- It is a First Public Working Draft. W3C says publication does not imply endorsement and that the document may be updated, replaced or made obsolete.
- The announcement does not list browser implementations, so do not assume any browser supports it.
- The pointer() default axis and the initial value of the named-timeline axis differ, so set the axis explicitly.
- Effects tied to a pointer do nothing for keyboard users and behave differently on touch screens. Sources were read on 26 November 2026. Products, rules and standards change, so check the linked originals before you rely on any detail.

How to try the idea safely

1. Read the draft and note which parts you would rely on.
2. Feature-detect with @supports and keep a complete static design as the default.
3. Use it only for decoration; never hide content or controls behind pointer position.
4. Respect prefers-reduced-motion and switch the effect off.
5. File issues with the working group, using the specification code in the title, if your use case does not fit.

What it means for front-end engineers and designers

This is one to prototype, not to ship as a dependency. If you maintain a design system, decide now where pointer effects are allowed and what the fallback looks like, so a later browser release does not create inconsistent behaviour across your products.

Frequently asked questions

- **What are CSS pointer-driven animations?** Animations whose progress follows the pointer’s position over a range, defined in a W3C draft.
- **Can I use them today?** The sources we read do not confirm support in any browser. Check current compatibility data and keep a fallback.

- **How are they related to scroll-driven animations?** They use the same progress-timeline approach, with pointer position instead of scroll position.
- **Which functions and properties are added?** pointer(), pointer-timeline-name, pointer-timeline-axis and the pointer-timeline shorthand.

Related reading

- What makes a good pull request?
- What is a feature flag?
- TPBOK standard

> Related credential: Certified Technical Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. W3C CSS Working Group. Pointer-driven Animations Module Level 1, First Public Working Draft. 24 November 2026. w3.org/TR/2026/WD-pointer-animations-1-20261124
2. W3C. First Public Working Draft: Pointer-driven Animations Module Level 1. 24 November 2026. w3.org/news/2026/first-public-working-draft-pointer-driven-animations-module-level-1

Published by IICTP, an independent technical credential body, on 26 November 2026. The current version of this post is at https://blogs.iictp.org/css-pointer-driven-animations-pointer-timeline-explained. More guides: blogs.iictp.org.