How Can You Use Performance Web APIs to Create Your Own Performance Measurement Tools?

While tools like Lighthouse and Chrome DevTools are great, sometimes you need custom insights tailored to your site.

That's where Performance Web APIs come in!

In this lesson, we'll break down how to use three key Web APIs to measure and analyze your website's speed.

But first, what are Performance Web APIs?

Performance Web APIs let developers track how efficiently a webpage loads and responds, directly from code. These APIs allow you to measure page load times, track rendering and interaction delays and analyze JavaScript execution time.

With these APIs, you can build your own performance monitoring tools without relying on third-party software!

Let's explore three powerful Web APIs you can use today.

First up, `performance.now()`.

This API gives you high-precision timestamps (in milliseconds) to measure how long different parts of your site take to load.

Let's say you want to measure how fast a function runs:

```
const start = performance.now();
// Run some code here
const end = performance.now();

console.log(`Execution time: ${end - start}ms`);
```

This is more accurate than using `Date.now()` because it measures time in milliseconds with microsecond precision, avoiding clock drift issues. You can use it to track script execution time, event response delays, and animation performance.

Next, the Performance Timing API.

This API gives you a breakdown on every single stage of page loading, from DNS lookup to `DOMContentLoaded`.

Want to measure how long your page takes to fully load?

```
let [navigationTiming] = performance.getEntriesByType("navigation");

if (navigationTiming instanceof PerformanceNavigationTiming) {
  // Calculate time from navigation start to DOM content loaded
  const pageLoadTime =
    navigationTiming.domContentLoadedEventEnd - navigationTiming.startTime;

  console.log("DOM Content Loaded Time:", pageLoadTime, "ms");
}
```

Key metrics you can track with this API are DNS lookup time - or in other words the connection speed, Time to First Byte (TTFB) - or server response speed, and `DOMContentLoaded` - or in other words, when the page is ready for interaction.

If your page load times are slow, this API pinpoints exactly where the delay happens!

And finally, let's talk about `PerformanceObserver`.

This API listens for performance events such as layout shifts, long tasks, and user interactions.

Want to monitor long-running JavaScript tasks?

```
const observer = new PerformanceObserver((list) => {
  list.getEntries().forEach((entry) => {
    console.log(`Long task detected: ${entry.duration}ms`);
  });
});

observer.observe({ type: "longtask", buffered: true });
```

And what can this API track? Well, it can track long tasks - or in other words, JavaScript that blocks rendering, layout shifts to detect UI jank, and First Input Delay (FID) - or how fast a page responds to user input.

If you want real-time performance tracking, this API is a game-changer.

So, which API should you use? Here's a quick comparison:

Performance API	Best For
`performance.now()`	Precise timing of functions and scripts
Performance Timing API	Measuring full page load performance
Performance Observer	Real-time monitoring of interactions and rendering

By combining these APIs, you can build your own performance measurement tools and track exactly what matters for your site most.

Questions

What is the main advantage of using `performance.now()` over `Date.now()`?

It's more accurate and measures time in milliseconds with microsecond precision.

Correct!

It works only on mobile devices and is not supported on desktops.

It measures system memory usage, including the browser's memory footprint.

It helps with CSS animations.

Which API can track long JavaScript tasks that slow down your site?

Performance Timing API

`Performance.now()`

Performance Observer

Correct!

PageSpeed Insights

What does the Performance Timing API help measure?

Individual JavaScript function execution time.

Full page load performance, including DNS lookup and TTFB.

Correct!

User interactions like button clicks.

The number of images on a webpage.

Navigated to How Can You Use Performance Web APIs to Create Your Own Performance Measurement Tools?