

How Does Performance Impact Sustainability?

(🔗) freecodecamp.org/learn/front-end-development-libraries-v9/lecture-understanding-performance-in-web-applications/how-does-performance-impact-sustainability



Improving web performance is crucial for delivering a fast and efficient user experience. But did you know that optimizing your website's performance doesn't just improve speed — it also helps the environment?

Every time a page loads, energy is consumed. And when sites are inefficient, they use more resources, leading to higher carbon emissions.

Let's break down how performance impacts sustainability and show you practical ways to optimize your code for a greener web.

The internet accounts for around 2% of global carbon emissions — that's the same as the airline industry! Every byte transferred requires electricity, from data centers to user devices. Larger files and inefficient scripts mean more power consumption. A high-performance website isn't just faster, it also reduces unnecessary processing and energy use.

Here's an example of inefficient JavaScript:

```
// Inefficient JavaScript
function fetchData() {
    for (let i = 0; i < 1000000; i++) {
        console.log("Processing...");
    }
    fetch("https://api.example.com/data")
        .then(response => response.json())
        .then(data => console.log(data));
}
fetchData();
```

This script runs excessive console logs, wasting CPU cycles and energy.

Now, let's optimize it:

```
// Optimized JavaScript
async function fetchDataOptimized() {
    const response = await fetch("https://api.example.com/data");
    const data = await response.json();
    console.log(data);
}
fetchDataOptimized();
```

By removing unnecessary loops and using `async/await`, we lower processing time and energy consumption.

Here are some practical optimization tips you can use in your future projects.

1. Minify Your Assets: Use tools like Terser or UglifyJS to shrink JavaScript and CSS files.
Smaller files load faster and require less energy.
2. Optimize Images: Use next-gen formats like WebP instead of PNGs or JPEGs to reduce file size without quality loss.
3. Reduce Network Requests: Each request adds overhead. Combine CSS and JavaScript files and use lazy loading to reduce load times.
4. Use Efficient Algorithms: A poorly optimized loop can increase CPU usage. This is the same as writing inefficient JavaScript that we saw in the example previously.
5. Leverage Caching & CDNs: Caching prevents repeated downloads, and CDNs deliver content closer to users, reducing transmission energy.

And now, what if you want to measure your impact and just how green your site is?

Use tools like Google Lighthouse to analyze performance and estimate energy savings. Or, for a deeper sustainability check, try the Website Carbon Calculator — as it estimates your page's environmental impact.

In conclusion, by understanding and implementing these performance fundamentals, developers can create web applications that are not only faster and more responsive but also more sustainable.

Questions

Which of these is not a way to optimize your assets?

Minify Your Assets.

Reduce Network Requests.

Use Light Mode.

Correct!

Leverage Caching & CDNs.

What percentage of global carbon emissions does the internet account for?

2%

Correct!

98%

99%

3%

What happens every time a page loads?

Energy is consumed.

Correct!

Energy is released.

Energy is created.

Energy is drained.

Navigated to How Does Performance Impact Sustainability?