Webinar #1 handout

Pick **JS**

THIS HANDOUT COMPLEMENTS THE WEBINAR 'JAVASCRIPT PERFORMANCE AT SCALE'. KEEP IT AROUND IN CASE YOU NEED IT, AND PICK AN ALTERNATIVE - **PICK JS!**

1. Consolidate and re-use code where it makes sense. In Adobe's case, this meant consolidating multiple libraries into a single library written from scratch. In your case, this may mean making more deliberate decisions when configuring code splitting with bundlers like Webpack or Parcel.
2. Consolidate requests where possible. This often entails creating a single endpoint that handles disparate data in a single request/response. GraphQL is one option for addressing this.
3. Consider removing dependencies. While you will lose the amazing benefits of community-backed libraries, you will be able to trim extraneous features or unnecessary browser support. You may be more effective in consolidating code, like utilities. In some cases (like jQuery), you may be able to switch to using native APIs.
4. Remove code for unsupported browsers. If you have changed your target browsers since code was written, you may have code that is no longer relevant.
5. Scrutinize cookies and their size. Cookies are usually sent on every request, regardless of whether it's a data or static asset request. Because of this, micro-optimizations may be worth their cost.
6. Leverage scope hoisting if possible to cut down on file size and improve performance.
7. Asynchronously load scripts if it helps users access useful content more quickly. If you're creating a standalone library for other websites to use, architect it in such a way that it can be asynchronously loaded and still useful.
8. Use a CDN like Akamai to get content closer to the user's physical location.
9. Over 1.3 billion people live in China. Consider acquiring a CDN presence within the China firewall.
10. Measure performance over time. Which metrics are important to your users? Use realistic test scenarios and try to maintain a consistent testing environment. Automate reporting and monitor changes in performance over time.

PREPARED BY

Aaron Hardy

Senior Computer Scientist
& Developer Advocate
at Adobe

He works on JavaScript that runs billions of times each day on the world's largest websites handling identification, analytics, personalization, and more. Aaron has a passion for open source and platforms and is an advocate in the development community. He loves technology, family and cycling and is an avid student of documentaries, podcasts, and YouTube.

/aaronhardy

/aaronius