System health benchmarks

Simulating the World Wide Web's activities in 90 minutes

nednguyen@
Last updated 09/26/2016
Shortlink: go/csh-benchmarks

Mailing lists: internal
Backlog: internal tasks tracking.
Tracking sheet: System health stories tracking

Benchmarking the browser is hard. There have been a lot of attempts to construct a comprehensive set of operation to test the browser' speed, but none of them reflects actual usage on the web, not even a close approximation. As the web grows more and more complex, we believe the best way to benchmark the browser is to run it through realistic user workflows against real web sites.

The Chrome System Health benchmarking effort aims to create a common set of user stories[1] on the web that can be used for all Chrome speed projects. Our benchmarks mimic average web users' activities and cover all major web platform APIs & browser features.

The web is vast, the possible combination of user activities is endless. Hence, to get a useful benchmarking tool for engineers to use for preventing regressions, launches and day to day work, we use data analysis and work with teams within Chrome to create a limited stories set that can fit a time budget of 90 minutes[2] machine time.

These are our key cover areas for the browser:

- Different user gestures: swipe, fling, text input, scroll & infinite scroll.
- Video
- Audio
- Flash
- Graphics: css, svg, canvas, webGL
- Background tabs
- Multi-tab switching
- Back button
- Follow a link
- Restore tabs

[1] User story is the atomic unit of benchmarking. Think about it as a single test case in a test suite (a system health benchmark).
[2] For detailed reason on why we have the hard limit of 90 minutes machine time, see this discussion about current automation lab's capacity (Googler only).

- Reload a page
- …. (Full tracking sheet)

Success to us means system health benchmarks create a wide enough net that catch major regressions before they make it to the users. This also means performance improvement to system health benchmarks translates to actual wins on the web, enabling teams to use this benchmark for tracking progress with the confidence that their improvement on the suite matters to real users.

To achieve this goal, just simulating user's activities on the web is not enough. We also partner with progressive-web-metrics team to track key user metrics on our user stories.

FAQ

Why are there multiple system health benchmarks?
Mobile web is very different from desktop web. Hence, to clearly distinguish between the two, we separate the system health benchmarks into mobile & desktop. The mobile version won't run on desktop machines, and vice versa.

In addition, since benchmarking memory usage has very high overhead, we also separate the memory benchmark from the rest even though they share the same user stories (equivalent user simulations on same web sites).

Finally, WebView's startup time is also a case we care a lot about, but since we don't have good startup metrics for Chrome desktop & Android, we temporarily create a separate benchmark for it.

So in total, we have 5 system health benchmarks:

- system_health.memory_mobile (WebView & Android)
- system_health.memory_desktop (Win, Mac & Linux)
- system_health.common_mobile (WebView & Android)
- system_health.common_desktop (Win, Mac & Linux)
- system_health.webview_startup_multiprocess (Webview)

How are system health benchmarks different from other chromium benchmarks?
System health are objective benchmarks that aim to *measure the ground truth*, detecting general regressions across Chrome. For that reason, we distance ourselves from the sort of benchmarks that a team might add for themselves to track *a specific feature*. For further clarifications, see this doc.

How do I contribute more user stories for system health benchmarks?

See this guide.