Ecosystem Infra Team Charter

July 2019
https://bit.ly/ecosystem-infra
foolip@ lukebjerring@ rbyers@

Mission

Establish the infrastructure and culture of testing for the web ecosystem to ensure a predictable experience

“Predictability” changes meaning in different contexts.

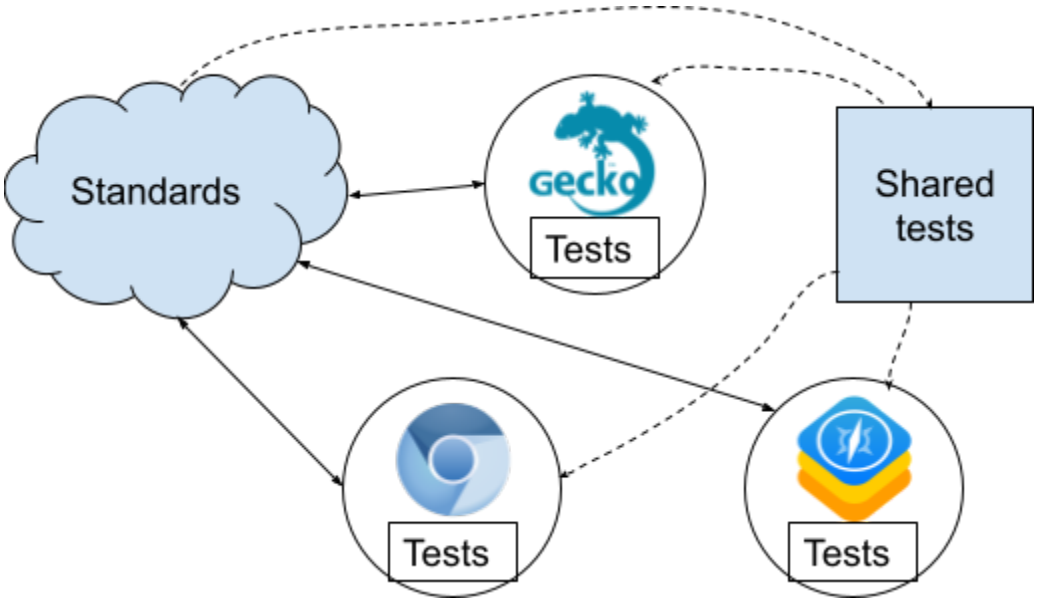
For web developers, predictable means they should be able to develop their applications and have them behave as expected, on all browsers.

For the end users, predictable means they should be able to browse the web and have a consistent experience, regardless of the device and browser they are using.

Background

We all agree that, *ideally*, implementations should match standards and vice versa.

Web platform implementation changes are mostly mediated by changes to the corresponding formal specification text. When implementers notice the changes, they will either implement as-is, or ask for further changes. Historically, sometimes there were shared tests, but often not.



This process is brittle, and mishaps are common. Key shortcomings:

- Changes to standards are rarely reviewed by all implementers. (Like with code review, there are diminishing returns, and everyone is busy.)
- After the change has been made, each implementer must notice and review it before knowing if it affects them. Therefore, changes easily go unnoticed.
- Implementers write unshared tests, which can cause differences between implementations go unnoticed. It is also a large duplication of effort.
- Failures in shared test suites are very hard to prioritize. Even new failures cannot always be traced back to a spec change.

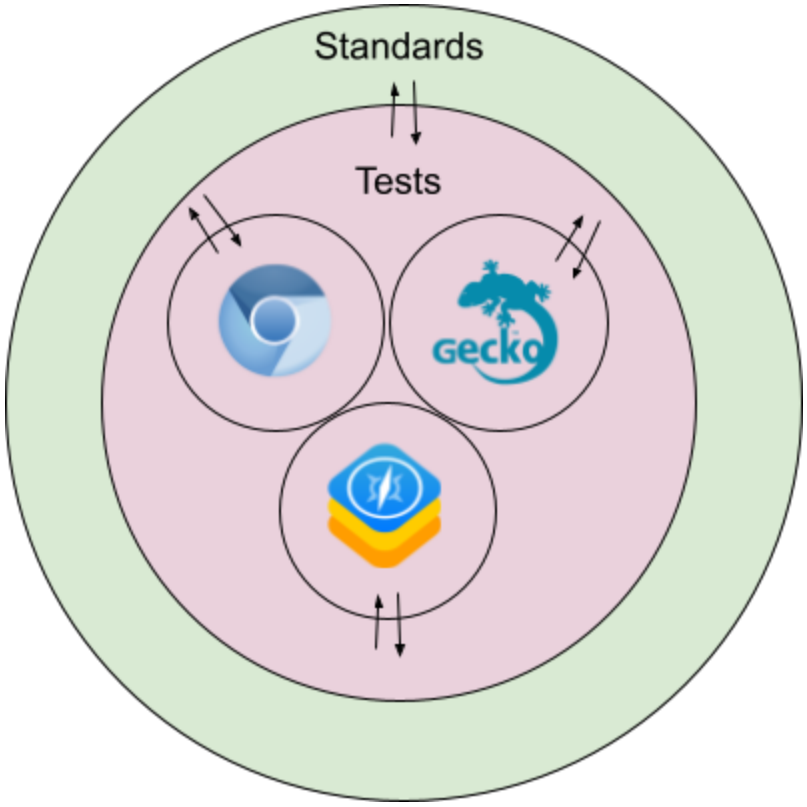
When things go awry, nobody in particular is responsible.

Vision

The web platform is engineered as a single, cohesive product.

The web platform is one of the largest software engineering projects ever, consisting of countless standards and implementations, all evolving over time. There's no single project manager, no single gatekeeper, but we are mostly held together by the standards process.

We envision that, as a natural outcome of our infrastructure and testing culture, web platform engineers are empowered by to make platform-wide changes with low friction; a change to any one part of the ecosystem reliably propagates to all others.



The top priority is to have real-world interoperability of implementations; standards and tests are an important means to that end.

We expect that:

- High-quality shared test suites are created and maintained in parallel with standards
    - Test results can be relied upon to be up-to-date and have good coverage.
- When we change standards, we seek feedback and buy-in from implementers.
    - If implementers do not eventually follow, we revisit the issue periodically until resolved.
- When we change web-exposed behavior in implementations, we ensure that standards and tests are updated to match, and revisit if necessary.
- No standard, test suite, or implementation is left unmaintained; making changes is always possible.
    - (Exception: parts of the web platform that are in the process of being removed.)

Standards

Standards are well-maintained.

We currently have poor insight into spec maintenance. Therefore, the team aims to:

- Maintain a list of all standards that are implemented, from Web IDL and other sources.

- Ensure that all standards link to their source and tests (example: standard, source, tests).
- Provide statistics on activity, responsiveness, test suite health, etc. This will assist the team in demarcating what constitutes “the current web platform”, further informing what is of interest to track.

Implementations

Implementations are consistent and spec-compliant

Implementations, and interoperability thereof, should be able to be measured, and compared.

For each browser engine:

- Automatic and frequent import and export of the shared tests, so that there is little overhead in writing shared tests together with implementation changes.
- The same continuous integration, with cross-browser results, as in upstream.

In many cases, it is also possible to directly compare implementations from source code or runtime behavior:

- Web IDL in different implementations (as well as standards and tests).
- Any other APIs exposed at runtime can be compared.
- HTML/SVG elements, attributes and CSS properties and values can be compared.

For these and similar cases, any new differences can be flagged/prevented, and existing issues fixed one way or another. While many may be surface-level differences, they can point to deeper issues, and avoiding such differences provides a baseline of interoperability.

Testing

Engineers actively collaborate on shared test suites with low friction.

Make it easy and delightful to work with shared tests, primarily web-platform-tests. Make contributing to shared tests an integrated and expected part of the standards and implementation workflow.

For web-platform-tests and potentially other test suites:

- Continuous integration; i.e., testing every change against many browsers and ensuring that failing or flaky tests are only introduced when intentional.
- A results dashboard for many browsers that is continuously updated, used to gauge the current health of the test suite and to prioritize work.
- Debuggability, i.e., the ability to work backwards from a failing test to the root cause.
- Support the shared infrastructure for test automation.
- Support, and potentially contribute to, high priority test suites.

For each standard:

- Standardized test automation APIs. This is feature-dependent but required whenever the web-exposed APIs themselves are not enough.

OKRs

- Ecosystem Infra 2018 OKRs
- Ecosystem Infra 2019 OKRs

Quarterly OKRs are linked from the yearly ones.

Project and work areas

WPT

web-platform-tests is a shared repository of tests for implementations of the web platform.

ecosystem-infra@ maintain a web-platform-tests 2-way sync in Chromium, while Mozilla maintain a 2-way sync in gecko and work has been done on 2-way sync for webkit by Bocoup.

ecosystem-infra additionally aim to expand and improve the capabilities of test-automation in WPT, including:

- Improvements to the ergonomics of the test suite itself
- Documentation and tutorials for new, and existing, contributors
- Experimenting with alternative testing frameworks (e.g. Puppeteer)

wpt.fyi

wpt.fyi is the web-platform-tests dashboard. It is an endpoint for the consumption of WPT results reports. results-collection (maintained by Bocoup) is a project for executing WPT, and collecting the results for publication to wpt.fyi.

There are custom checks for PRs (pull requests) in wpt, used a feedback system in detecting and reporting unintentional regressions. Additionally, web-platform-tests.live supports mirroring of PRs for live testing.

A structured search of the test results allows for identifying specific failures.

web-confluence

The Web API Confluence Dashboard crawls and reports the surface area of Web Platform APIs.

Other

Bikeshed is a spec compiler, which helps editors be more productive, and so can contribute to well-maintained standards. However, it isn't the only spec production tool, any tooling we create must consume the output and not assume Bikeshed.

html-build/wattsi is similar to Bikeshed.

Project backlog

See Ecosystem infra project backlog for a (wide) range of project ideas.

Bug queries:

- component:Blink>Infra>Ecosystem
- https://github.com/w3c/web-platform-tests/labels/infra