What’s Up With Web Standards

With special guest Chris

We've heard about the web platform, so what about web standards? How have they evolved, and what does the process look like now? Today’s special guest is Chris, who has worked all across the standards space, as is Google’s W3C AC representative.

Web standards

Web standards are guidelines established to ensure interoperability across different web browsers. Historically, multiple browsers needed to function similarly, but the absence of detailed standards led to issues that required reverse engineering. The primary goal of these standards is to ensure that the open web operates consistently across various platforms, without relying on a single codebase.

W3C, SDO and AC

The W3C (World Wide Web Consortium) is the largest Standards Developing Organization (SDO) for web standards, overseeing the development of technologies like HTML, DOM, and CSS. The Advisory Committee (AC) consists of representatives from member organizations, who vote on standards-related decisions. While it may seem powerful, the role mainly involves coordinating with others to determine voting positions. The Advisory Board, an elected group, provides guidance to the W3C on areas of focus, including technical trends like AI and logistical matters like health restrictions at meetings.

W3C members

The W3C has around 400 members, which include a mix of large companies (like Google, Microsoft, Amazon, IBM, Apple, and Mozilla), smaller companies, academic institutions, and invited experts. Invited experts contribute without being formal members or paying dues. Much of the W3C's work is public or semi-public, allowing participation without official membership. A key challenge for web standards is ensuring interoperability and securing intellectual property rights to prevent legal issues, such as lawsuits over implementing technologies like cascading style sheets. This involves coordinating with patent teams to ensure everything is legally sound and freely implementable.

W3C member responsibilities

As one advances in a field, the work often shifts from hands-on tasks to more administrative roles like running meetings and coordinating people. Despite this, there's still a desire to engage in technical work, such as contributing to the Web Audio Working Group at the W3C, where significant progress was made on powerful audio APIs for browsers. Although responsibilities may lean toward higher-level tasks, efforts are made to stay connected to technical topics.

The difference between "specifications" (specs) and "standards" is that specifications are documents that outline an idea and its development into a standard. The process begins with "explainers" that define the problem and potential solutions, gradually refining the approach with community input. The journey from spec to standard involves detailing how the solution should be implemented, resulting in a comprehensive guide. A standard is the finalized specification, providing a clear manual for others to implement the feature consistently across different platforms.

Standards contributors

Chrome implements web standards by adhering to specifications, even if those specs haven't yet become fully accepted standards. Most web platform code in Chrome is guided by a spec, ensuring that new features align with broader standards processes. While not everything reaches an interoperable standard immediately, there should always be a specification in place.

Various organizations contribute to the specs and standards implemented in Chrome. WHATWG handles HTML, DOM, and other core web technologies, especially after branching off from the W3C to focus on the web as an application platform. The IETF manages protocols like HTTP, while TC39 is responsible for JavaScript development. Khronos contributes to graphics work, influencing features like WebGPU. These organizations often collaborate, and the boundaries between their contributions can be fluid.

Occasionally, different organizations develop specs for the same feature, forcing browsers to choose which one to implement. However, this situation usually doesn't persist for long. One solution often gains traction in the marketplace or standard space due to the primary goal of interoperability.

In some cases, multiple features are developed in different places and aren't fully standardized across all browsers. For example, image formats, video, and audio codecs may vary slightly between browsers, with each supporting a similar but not identical set. These overlaps occur, but for most new features or functionalities, a community will typically form around one approach, leading to a more unified specification.

Standards and browsers

All major browser developers generally follow the same specifications and attend the same standards organizations. However, browsers may implement different features or make distinct choices based on their priorities. For example, some browsers might prioritize privacy, disabling features that could be used for fingerprinting, while others might focus on creating the best immersive web experience. These decisions can lead to variations in what each browser supports, but the overall goal of the web remains global interoperability.

Standards organizations like the W3C have members worldwide, covering Europe, Asia, and the Americas. While some regions may not be as involved in developing web engines, the core specifications are generally adhered to across different regions. The user interface elements of a browser, such as how favorites are organized, are often left to the discretion of the browser developers and are not strictly defined by standards.

Implementation of standards

For individual specifications, ensuring correct implementation involves rigorous testing. Modern standards, such as W3C recommendations, require comprehensive test suites to verify adherence to the spec. This emphasis on testing contrasts with practices from 20 years ago, when standards were less rigorously tested.

While specifications are clearly defined, there is no overarching meta-spec that dictates every aspect a browser must implement. Browsers make choices about which parts of specifications to include, and there isn't a one-size-fits-all checklist for compliance.

In W3C terminology, a "Recommendation" is the term used for what is considered a standard. Unlike other standards organizations, W3C prefers the term "Recommendation" to emphasize a less rigid, more evolving approach to standards development.

Browser engineers vs. Web developers

The push for new web features can come from various sources. Companies, particularly browser vendors, play a significant role, but contributions from web developers are also crucial. Browser engineering teams often need to be interested in and support the implementation of new features. While contributing code to open projects like Chromium, WebKit, or Gecko is possible, gaining traction can be challenging without clear priorities or a direct path for feature adoption.

Web developers can raise new feature ideas through incubation venues, such as the Web Platform Incubation Community Group at the W3C, or initiatives like the Web We Want. These

platforms allow developers to propose new features, which can then be considered by browser vendors and engineering teams.

The W3C comprises a diverse range of members, not limited to browser engineers. Many members focus on areas beyond browser specifications, such as decentralized digital identity. While some of these specs might eventually lead to browser APIs, they are not always directly related to core browser functionality.

Organizations like the W3C are increasingly addressing web use cases beyond traditional browser functionality. There is a growing recognition that the web encompasses more than just HTML delivered over HTTP and involves various web systems that may not necessarily be browsers.

The process of developing and implementing new web features can be complex and unpredictable, with no single path to success. Browser vendors like Google, which develops Chrome and also manages significant web properties such as YouTube, play a key role in this process. These web properties provide valuable feedback and collaboration opportunities, helping to refine and advance web standards. However, it is also important to seek input from a broad range of sources to avoid being confined to a narrow perspective.

The evolution of specs

Specs can evolve and change in several ways:

1. Updates to Existing Specs: Sometimes, a specification is updated to address new requirements or changes in technology. This might involve refining existing features or adding new ones to keep up with current use cases.

2. Deprecation: If a spec becomes obsolete or less relevant, it might be deprecated. This means it's no longer recommended for use, and support for it may eventually be removed.

3. Creation of New Specs: New use cases or technologies can lead to the development of entirely new specs. This can happen when existing specs no longer meet the needs of developers or when new functionalities are needed.

The process of evolving a spec is often lengthy and complex. For instance, a feature can take several years to move from initial idea to a fully accepted standard. This is due to the need for broad community consensus, extensive testing, and multiple iterations. An example is the evolution of client-side caching mechanisms. AppCache was initially introduced for caching web resources but proved problematic due to its limitations. It was eventually replaced by Service Workers, which offer more flexibility and control. Overall, evolving specs involves continuous feedback, collaboration among stakeholders, and adapting to changing technologies and requirements.

Key events in the evolution of web standards and interoperability

Mid '90s: The era of intense browser competition, often referred to as the "browser wars." This period saw significant differences in how browsers implemented web standards, leading to major compatibility issues.
2004: The Web Hypertext Application Technology Working Group (WHATWG) was established to address the need for more unified web standards. This group played a key role in evolving web standards.
Late 2000s: After about a decade of frustration with inconsistent browser implementations, efforts from groups like WHATWG began to result in more standardized approaches. The focus shifted to creating coherent specifications that could be adopted by all major browsers.
Today: While interoperability issues still exist, there's a more structured approach to resolving them. The standards community actively works to address and correct implementation differences. For example, issues found in CSS are often reported to the relevant spec and resolved relatively quickly.

Current Challenges in Web Standards

1. Feature Prioritization and Investment:

- **Divergent Priorities**: Different browser vendors prioritize different features, leading to discrepancies in implementation. For instance, while Google may invest heavily in certain features, other browsers may not, causing fragmentation in available features across browsers.
- **Community and Technical Reviews**: New features often require extensive feedback from various stakeholders, including standards organizations and technical groups, to ensure compatibility and broader adoption.

2. Device and Context Variation:

- **Feature Set Choices**: Different devices may implement varying subsets of web features based on their capabilities and intended use. For example, a high-end smartphone might support advanced features not available on an older or less capable device.
- **Context-Specific Browsers**: Some devices, like cars or older computers, may have simpler browsers optimized for specific tasks rather than a full suite of modern features.

3. Evolving Standards:

- **Standards Evolution**: Web standards and specifications continue to evolve, sometimes creating challenges for backward compatibility. As new features are added, older ones may be deprecated or replaced, which can lead to compatibility issues.

4. Legacy and Modernization:

- **Legacy Code**: The web's history includes many legacy features and codebases. Updating or removing these can be challenging due to the need to maintain compatibility with existing implementations and user expectations.
- **Modern Features**: Balancing the introduction of cutting-edge features with the need to support older or less capable systems can be complex.