API Deprecations and Removals in Chrome 56

By Joseph Medley
Technical Writer

In nearly every version of Chrome, we see a significant number of updates and improvements to the product, its performance, and also capabilities of the Web Platform. This article describes the deprecations and removals in Chrome 56, which is in beta as of December 8. This list is subject to change at any time.

Remove support for SHA-1 certificates

The SHA-1 cryptographic hash algorithm first showed signs of weakness over eleven years ago and recent research points to the imminent possibility of attacks that could directly impact the integrity of the web public key infrastructure (PKI).

To protect users from such attacks, Chrome no longer supports SHA-1 certificates starting in Chrome 56, whose stable release is in January 2017. Visiting a site using such a certificate results in an interstitial warning. We provide more details on the Chrome Security Blog.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Remove CBC-mode ECDSA ciphers in TLS

TLS's CBC-mode construction is flawed, making it fragile and very difficult to implement securely. Although CBC-mode ciphers are still widely used with RSA, they are virtually nonexistent with ECDSA. Other browsers still support these ciphers, we believe the risk is low. Additionally, ECDSA in TLS is used by few organizations and usually with a more complex setup (some older clients only support RSA), so we expect ECDSA sites to be better maintained and more responsive in case of problems.

TLS 1.2 added new ciphers based on AEADs which avoids these problems, specifically AES_128_GCM, AES_256_GCM, or CHACHA20_POLY1305. Although we are only requiring this for ECDSA-based sites at this time, it is recommended for all administrators. AEAD-

based ciphers not only improve security but also performance. AES-GCM has hardware support on recent CPUs and ChaCha20-Poly1305 admits fast software implementations. Meanwhile, CBC ciphers require slow complex mitigations and PRNG access on each outgoing record. AEAD-based ciphers are also a prerequisite for HTTP/2 and False Start optimizations.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Remove user gestures from touch scroll

We've seen multiple examples of poorly written or malicious ads that trigger navigation for touch scrolls either on `touchstart` or all `touchend` events. If a 'wheel' event can't open a pop-up, then touch scrolling shouldn't either. This may break some scenarios, for example, media not playing on touch, or pop-ups not opening on touch. Safari already silently fails to open pop-ups in all of these scenarios.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Disallow all fetches for scripts with invalid type/language attributes

Currently, Chrome's preload scanner fetches items in `<scripts>` elements regardless of the value of the `type` or `language` attribute, though the script will not be executed when parsed. By deprecating the fetch, the preload scanner and the parser will have the same semantics, and we will not be initiating fetches for scripts we will not use. This is intended to save data for users who navigate to sites with a lot of custom script tags that are post-processed (like `type="text/template"`, for example).

The use case of using invalid scripts to ping servers is adequately covered by the sendBeacon API.

This change aligns Chrome with Safari, though Firefox still requests scripts regardless of type or language.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Remove MediaStreamTrack.getSources()

This method is no longer part of the spec and is not supported by any other major browser. It has been replaced by **`MediaDevices.enumerateDevices()`**, which Blink has supported without flags since version 47 and which is also supported by other browsers. An example of this is shown below. This hypothetical **`getCameras()`** function first uses feature detection to find and use **`enumerateDevices()`**. If the feature detection fails, it looks for **`getSources()`** in **`MediaStreamTrack`**. Finally, if there is no API support of any kind return the empty **`cameras`** array.

```
function getCameras(camerasCallback) {
  var cameras = [];
  if('enumerateDevices' in navigator.mediaDevices) {
     navigator.mediaDevices.enumerateDevices()
      .then(function(sources) {
        return sources.filter(function(source) {
          return source.kind == 'videoinput'
        });
      })
      .then(function(sources) {
        sources.forEach(function(source) {
          if(source.label.indexOf('facing back') >= 0) {
            // move front facing to the front.
            cameras.unshift(source);
          }
          else {
            cameras.push(source);
          }
        });
        camerasCallback(cameras);
      });
  }
  else if('getSources' in MediaStreamTrack) {
    MediaStreamTrack.getSources(function(sources) {

      for(var i = 0; i < sources.length; i++) {
        var source = sources[i];
        if(source.kind === 'video') {

          if(source.facing === 'environment') {
            // cameras facing the environment are pushed to the front of the page
            cameras.unshift(source);
          }
          else {
            cameras.push(source);
          }
        }
      }
      camerasCallback(cameras);
```

```
    });
  }
  else {
    // We can't pick the correct camera because the API doesn't support it.
    camerasCallback(cameras);
  }
};
```

Intent to Remove | Chromestatus Tracker | Chromium Bug

Remove reflected-xss CSP directive

Early drafts of the Content Security Policy Level 2 spec contained a `reflected-xss` directive which offered nothing more than the `X-XSS-Protection` header other than a different syntax. This directive was removed from the spec in 2015, but not before it was implemented in Chrome. Support for this directive is now being removed.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Replace CSP 'referrer' directive

The CSP `referrer` directive allowed site owners to set a referrer policy from an HTTP header. Not only does this feature have very low usage, it has also no longer part of any W3C spec.

Sites that still need this functionality should use `<meta name="referrer">` or the new Referrer-Policy header.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Remove PaymentAddress.careOf field

The `PaymentAddress` interface has a `careOf` field which is non-standard (no well-known address standards support it). The `careOf` field is also unnecessary, the recipient and organization fields sufficiently support all necessary use cases. Adding `careOf` poses significant issues in terms of interoperability with existing postal address schemas and APIs. For a fuller discussion, read the spec removal proposal on GitHub.

Intent to Remove | Chromium Bug

Remove SVGViewElement.viewTarget

The `SVGViewElement.viewTarget` attribute is not part of the SVG2.0 specification and it's usage is small or nonexistent. This attribute was deprecated in Chrome 54 and has now been removed.

Intent to Remove | Chromestatus Tracker | Chromium Bug

Deprecation policy

To keep the platform healthy, we sometimes remove APIs from the Web Platform which have run their course. There can be many reasons why we would remove an API, such as:

- They are superseded by newer APIs.
- They are updated to reflect changes to specifications to bring alignment and consistency with other browsers.
- They are early experiments that never came to fruition in other browsers and thus can increase the burden of support for web developers.

Some of these changes will have an effect on a very small number of sites. To mitigate issues ahead of time, we try to give developers advanced notice so they can make the required changes to keep their sites running.

Chrome currently has a process for deprecations and removals of API's, essentially:

- Announce on the blink-dev mailing list.
- Set warnings and give time scales in the Chrome DevTools Console when usage is detected on the page.
- Wait, monitor, and then remove the feature as usage drops.

You can find a list of all deprecated features on chromestatus.com using the deprecated filter and removed features by applying the removed filter. We will also try to summarize some of the changes, reasoning, and migration paths in these posts.

Last updated July 2, 2018.