To inform a refactoring of the Permissions spec to handle temporary permissions and future permission models better.

See also:

- #86: Model temporary permissions better
- #90: Specify how request and revoke work on push and midi permissions.
- #94: Separate "request access" from "request permission"?

Three refactoring attempts:

- #95: Allow UAs to maintain zero or multiple permission stores. Also needs #91: Define an algorithm to update the permission storage.
- #96: Describe the permission store using constraints instead of a full model.
- #97: Give the UA permission to return anything it wants from query().

TOC

Behavior of existing permission UIs:

Geolocation

Firefox default

watchLocation() shows a bubble. If the user accepts, it generates a stream of locations.

A second watchLocation() shows another bubble. If the user accepts, it generates another stream of locations.

If the user subsequently blocks geolocation, the existing streams keep returning locations.

Safari default

watchLocation() shows a bubble. If the user accepts, it generates a stream of locations.

A second watchLocation() with the same global object does not show a bubble. Reloading the iframe causes a new global and hence a new bubble.

If the user subsequently blocks geolocation, the existing streams keep returning locations.

Firefox "Always", Safari "Remember for 1 day", Chrome default, Edge

watchLocation() shows a bubble. If the user accepts, it generates a stream of locations.

In Safari and Chrome, revoking the permission doesn't take effect until the user reloads the page. In Firefox, it causes the next watchLocation() call to show another prompt, even before a reload, but it doesn't interrupt the stream of locations.

I didn't see a way to revoke permission in Edge.

I didn't test Safari after 1 day to see exactly how it revokes its permission.

Camera

Firefox default

getUserMedia() shows a bubble. If the user accepts, it generates a video stream.

A second getUserMedia() shows another bubble. If the user accepts, it generates another video stream.

If the user subsequently blocks camera access, the existing streams keep returning video.

The user can stop sharing a camera to a single tab, which stops it for all frames in that tab.

Firefox "Always"

Same as above, but subsequent requests in any tab don't show a bubble. If the user revokes any tab's camera, the next request does show a bubble.

Safari has no getUserMedia()

Edge

getUserMedia() shows a bubble. If the user accepts, it generates a stream of locations.

A second getUserMedia() with the same global object does not show a bubble. Didn't check reloading the iframe.

Didn't see a way to revoke permission.

Chrome

getUserMedia() shows a bubble. If the user accepts, it generates a video stream.

Subsequent calls in any tab generate video streams with no bubble.

If the user revokes, they have to reload before there's any effect, even in the same tab.

Origin scoping

When a permission is granted persistently through a request bubble, it's only granted to the origin that requested it, not to any other origin, including ones with the same public domain suffix. Chrome allows certain permissions to be wildcarded through enterprise policy and a settings UI, but not through a page's request.

- Firefox scrubs port numbers out of origins when granting permissions. May allow subdomains. Denying permission does a lot of cleaning. May go all the way up to the public suffix when denying.

Policy

Enterprise policy and extensions can grant or deny permission to particular origins without explicitly asking the user's permission.

Future permission possibilities

persistent-storage

Wants to require that if the user grants permission to one tab, it's granted to all other tabs and survives past reload. If the browser doesn't want to allow such persistent permission, it has to deny any tab's request.

Media streams

Wants to require that a frame can't have access to a camera/mic unless the user has granted access to that frame's origin **in the context of** the top-level origin.

Auto-deny

Chrome wants to collect metrics on how users interact with permissions and avoid explicitly asking the user when we think we know their answer. For example, if a site has an unusually high revoke rate for notifications, we might prevent it from even asking users.

Auto-grant

Chrome auto-grants midi-without-sysex.

Do we want to allow UAs to auto-grant more sensitive permissions like persistent-storage, background-sync, notifications?

Note: Some browsers may auto-grant WebRTC-related permissions.

Permission delegation

https://noncombatant.github.io/permission-delegation-api/ proposes that an iframe might get its permissions delegated from its parent frame. If a capability isn't listed in an iframe's [permissions] attribute, that capability would be "denied". If it is listed, it would be "prompt" or "granted" matching the value for its parent frame, even if the parent frame has a different origin. Granting permission to the iframe persistently would grant it to the top-level frame's origin.

midi and push

The "midi" and "push" permissions are actually 2 related permissions each: "midi without sysex", "midi with sysex", "push with a notification" and "push without a notification". The query() API wants to spell these as {name: "midi", sysex: true}, etc., but we can have nearly independent choices of "granted" or "denied" for sysex:true vs sysex:false.

Feature Policy

https://igrigorik.github.io/feature-policy/ proposes a CSP-like header to let resources disclaim use of particular features, which are a superset of the things we ask permission for.